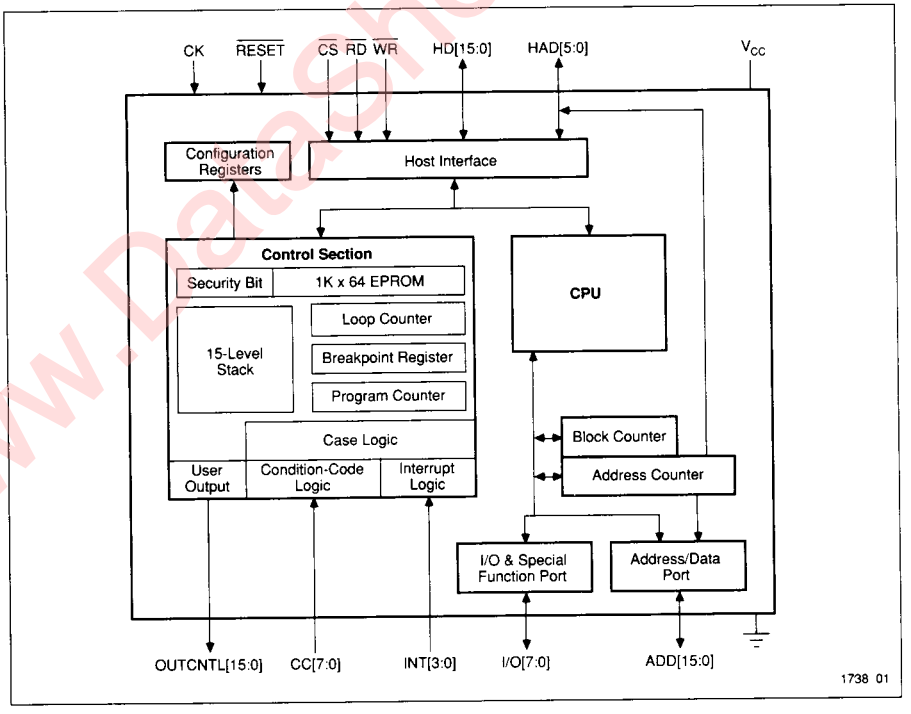
Programmable Peripheral
PAC1000
User-Configurable Embedded Controller

Preliminary

Features

- High-Performance User-Configurable Embedded Controller
  - 16 MHz Instruction Execution, Output Port, and Address Bus
- Single-Cycle Control Architecture
  - One Cycle Per Instruction
- 16-bit CPU
  - Arithmetic Operations, Logic Operations, 33 General-Purpose Registers
- Address Generation
  - Up To 4 Mbytes Address Space
- High-Level Development Tools – System Entry Language, Functional Simulator, and Device Programmer
- Re-Programmable Program Store
  - 1K x 64-Bit EPROM for CPGA Package
  - 1008 x 64-Bit EPROM for PQFP Package
- Re-Programmable Program Store
  - On-Board 1K x 64-Bit EPROM
- Two Operating Modes
  - Host Processor Peripheral or Stand-Alone Controller
- Security
  - For EPROM Program Memory
- Package Availability
  - 88-Pin Ceramic PGA and 100-Pin PQFP

Figure 1. PAC1000 Block Diagram



4

General Description

The PAC1000 User-Configurable Embedded Controller is based upon an architecture that enables it to execute complex instructions in a single clock cycle. Each PAC1000 instruction can perform three simultaneous operations: Program Control, CPU functions, and Output Control, as shown in Figure 2. The PAC1000 can also perform address generation or event counting simultaneously with instruction execution. The PAC1000 is also capable of performing a conditional test on up to four separate conditions and multi-way branching in a single cycle.

The PAC1000, with its System Development Tools, matches the development cycle and ease of use of any standard microcontroller.

The high performance and flexibility of the PAC1000 were previously available only to designers who could afford the long development cycle, high cost, high power, and large board space requirements of a building-block solution (i.e., Sequencer, Microcode Memory, ALU, Register File, PALs, etc.)

The unique capabilities of PAC1000 are easily utilized with System development tools, which include a PACSEL C-like System Entry Language, a PACSIM Functional Simulator, and a MagicPro™ Device Programmer. All System Development Tools are PC-based and will operate on an IBM-XT, AT, PS2 or compatible machine. For more information, contact your nearest WSI sales office or representative.

Figure 2. Single-Cycle Control Architecture

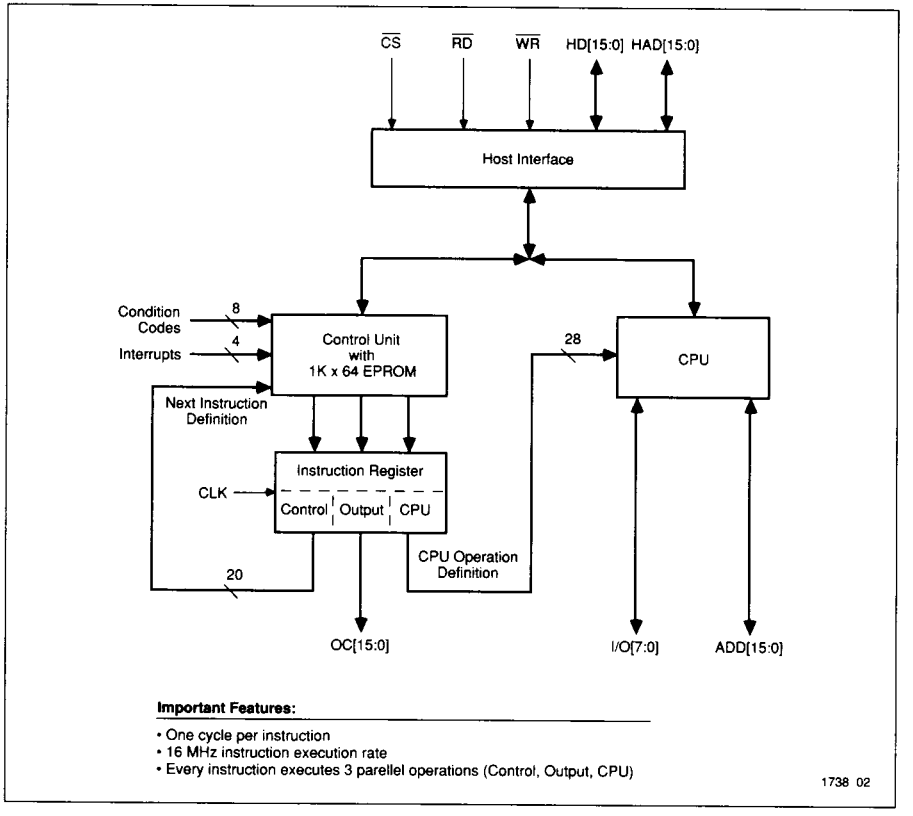


Table 1. Pin Description

Signal	I/O	Description
HD[15:0]	I/O	*Host Data.* PAC1000 Data I/O Port via the Host Interface. Can also be configured to generate 16-bit address or status. Can serve as a general-purpose Data I/O Port.
HAD[5:0]	I/O	*Host Address.* Can be configured to output the lower six bits of the 22-bit Address Counter; can be used as a Host Interface function address, or as a general-purpose 16-bit port.
$\overline{CS}$	I	*Chip Select (active low).* Used with $\overline{RD}$ and $\overline{WR}$ to access the device via the Host Interface.
$\overline{RD}$	I	*Read Enable (active low).* Used with $\overline{CS}$ to output Program Counter, Status Register, or Data Output Register to HD[15:0] bus lines.
$\overline{WR}$	I	*Write Enable (active low).* Used with $\overline{CS}$ to write HD Bus data via the Host Interface into the PAC1000 FIFO.
CK	I	*Clock.*
CC[7:0]	I	*Condition Codes.* Condition-code inputs for use with Call, Jump, and Case instructions.
INT[3:0]	I	*Interrupts.* General-purpose, positive-edge-triggered interrupt inputs.
$\overline{RESET}$	I	*Asynchronous Reset (active low).* Resets Input/Output registers and counters, tri-states all I/O, and sets the Program Counter to 0.
OUTCNTL[15:0]	O	*Output Control.* User-defined Output Port. May be programmed to change value every cycle.
ADD[15:0]	I/O	*Address Port.* Outputs data from Address Counter or Address Output Register when configured as an output. When configured as an input, reads data to Address Input Register.
I/O[7:0]	I/O	*Input or Output Port.* Individually configurable bidirectional bus. As simple I/O, outputs come from the I/O Output Register, and inputs appear in the I/O Input Register. As special I/O functions, provides status, handshaking, and serial I/O. Alternatively, these signals can be used to extend the OUTCNTL or ADD lines.

Architectural Overview

The PAC1000 is a user-configurable embedded controller optimized for high-performance control systems. The primary architectural elements, shown in Figure 3, are the Control Section, 16-bit CPU, Host Interface, 16-bit Address Port, 16-bit Output Control, 8-bit I/O Port, and Configuration Registers.

The PAC1000 can be used as a stand-alone embedded controller or as a peripheral to a host. In the latter case, the Host Data (HD) and Host Address (HAD) buses, together with the $\overline{CS}$, $\overline{RD}$, and $\overline{WR}$ pins allow for direct connection to a host bus. User-defined commands to the Control Section or data to the CPU can be loaded through the Host Interface.

In the stand-alone mode, the Host Interface ports can be used as additional address, data or I/O ports using the Data Output Register (DOR) and Data Input Register (DIR). The ADD port can be used to generate addresses through the Address Output Register (AOR) or the Address Counter. A DMA channel can be formed on the Host Interface using these and the Block Counter (BC) register. In addition, the ADD port can be used as a data bus or an I/O port, depending on how the chip is configured. Each pin in the I/O port can be configured individually as input, output, or special function. The special functions allow the control of internal PAC1000 elements (counters, I/O buffers) by other board elements.

The 16-bit CPU is highly parallel and can operate on operands from the 32x16-bit register file, miscellaneous register (AOR, AIR, DOR, DIR, Q, etc.), or constants loaded from the internal program-store EPROM.

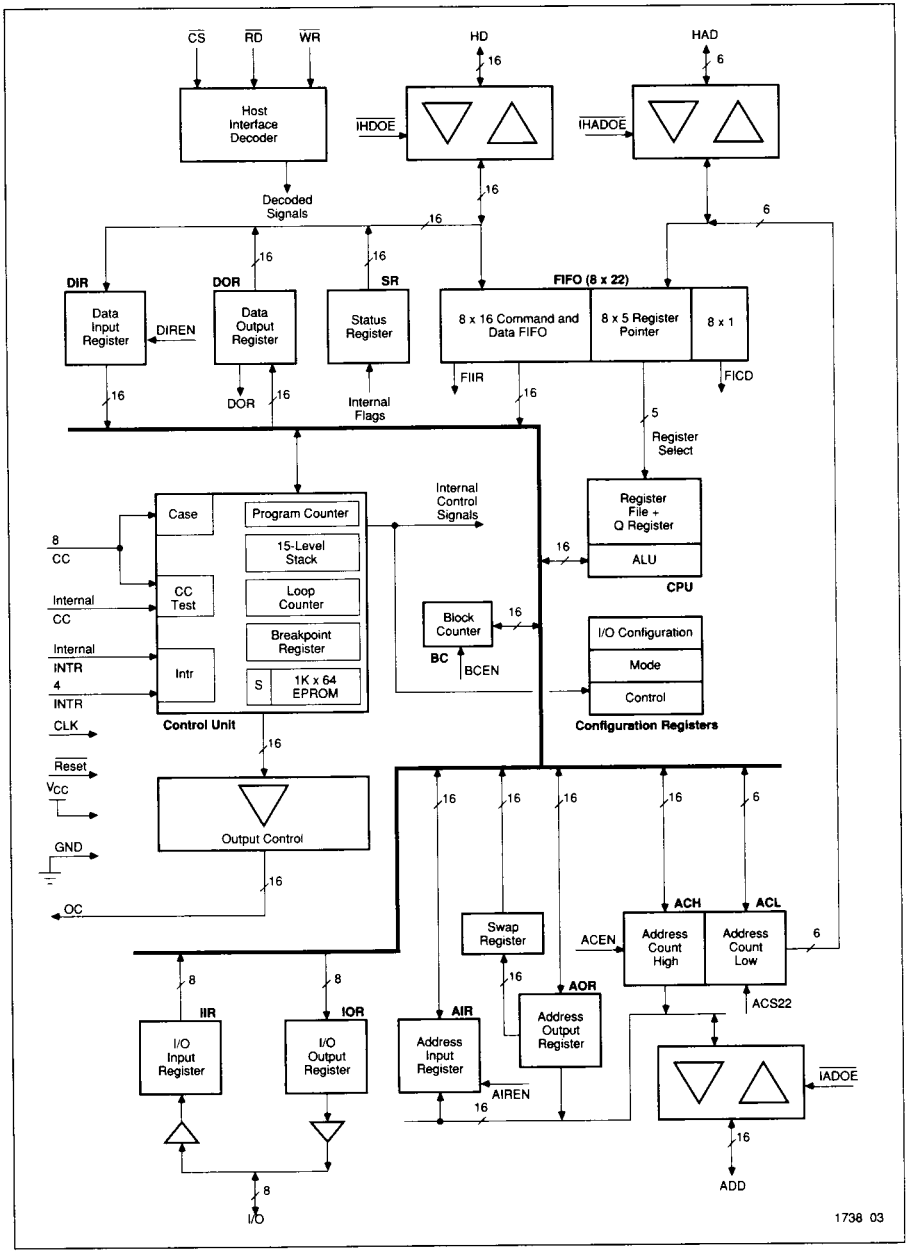
The internal and external operations of the PAC1000 are controlled by the Control Section. The 16 Output Control (OC) lines are general-purpose outputs. Each of them can be changed independently every clock cycle. They provide a very fast means to control various processes outside the chip.

In every clock cycle, one instruction is executed. Each instruction consists of up to three operations in parallel:

- Instruction Fetch—the next instruction is fetched from the 1Kx64 EPROM by the Program Control.
- Execution—the CPU executes an instruction.
- Output—placed on the Output Control (OC) lines.

Program flow can be changed through the condition-code inputs in one clock cycle or through the interrupt inputs after two clock cycles. Single-cycle 16-way branches can be done using the Case instruction, which samples four condition codes per cycle. Nested loops and subroutines can be carried out with the 15-level stack and the loop counter. The chip configuration can be changed in any cycle by loading the Configuration Register using the Program Control instruction portion.

Figure 3. Detailed Block Diagram



Operational Modes

The two basic modes of operation for the PAC1000 are either as a memory-mapped peripheral (Figure 4) or as a stand-alone controller (Figure 5).

In the peripheral mode, the host processor can asynchronously interface with the PAC1000.

Figure 4. Peripheral Mode

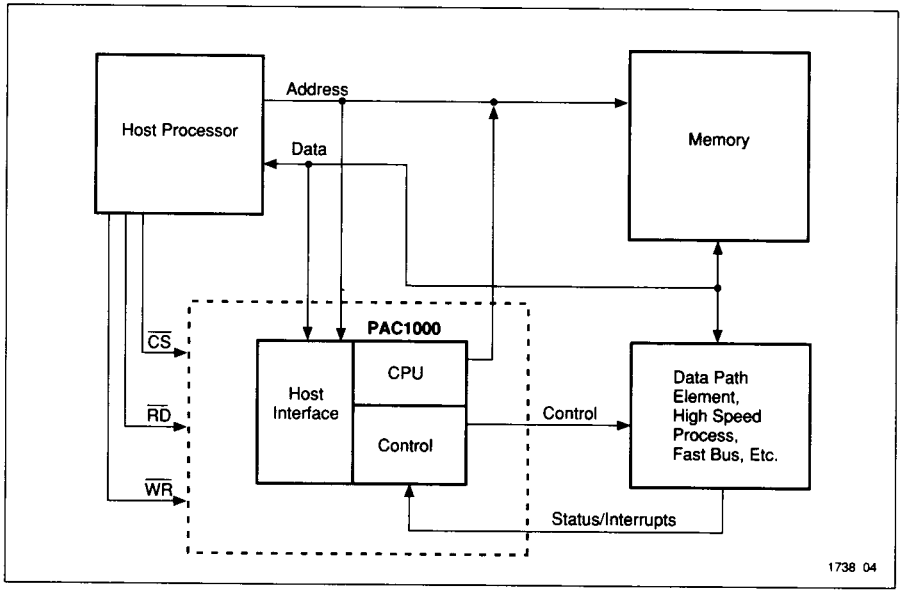
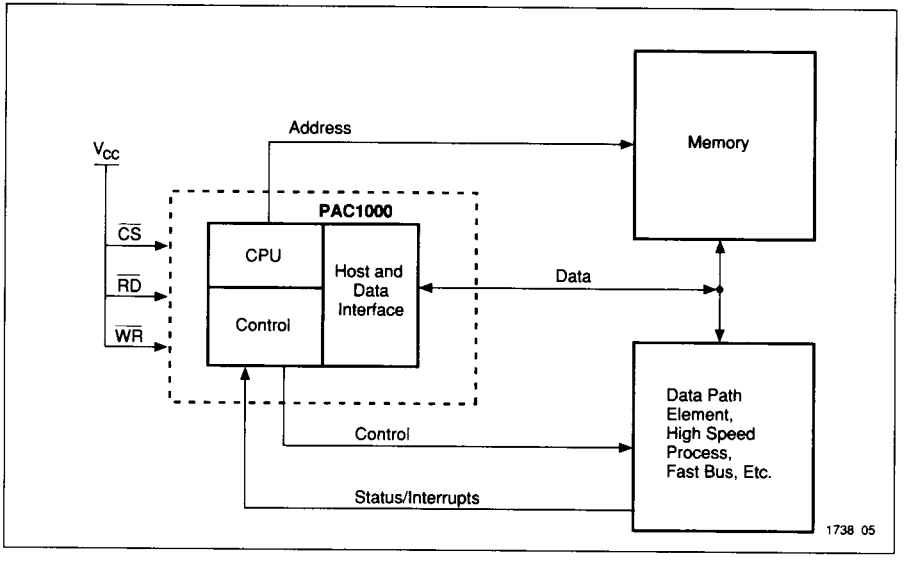


Figure 5. Stand-alone Mode



Host Interface

The Host Interface section of the PAC1000, shown in Figure 6, includes the Input Command/Data FIFO, Input/Output Data Registers, and the Status Register.

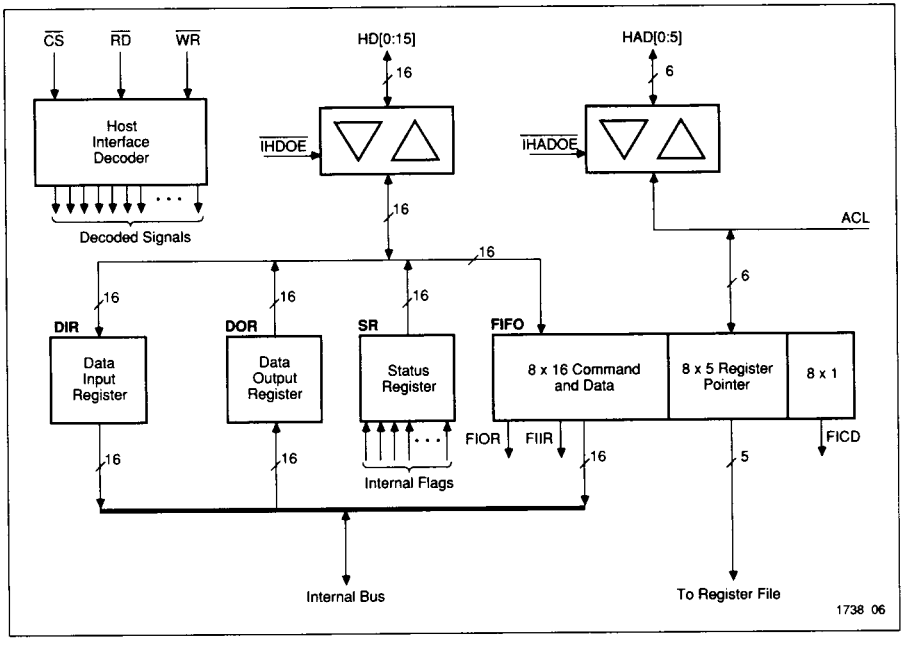
FIFO

When the PAC1000 serves as a peripheral to a host, the FIFO is used to asynchronously load commands or data into the PAC1000. In order to write into the FIFO, $\overline{CS}$ and $\overline{WR}$ must have low-to-high transitions. The information written into the FIFO is specified by the 16-bit Interface Data bus (HD) and the 6-bit Host Address bus (HAD). Since the FIFO is used only to buffer data and commands from a host, it is inoperative when the PAC1000 is in stand-alone mode.

Bit five of the HAD bus specifies whether the input to the FIFO is command (HAD5=1) or data (HAD5=0). HAD5 is connected to the FICD internal Condition Code that can be sampled by the Control Section. If a command is written, then the lower 10 bits of the HD bus are used as the branch address for one of the 1024 locations in the Program Memory EPROM. At that location a user defined command or subroutine should exist which executes the needed operation. If the information is data, then the lower 5 bits of the HAD bus specify which CPU register is to be loaded from the HD bus.

This method of operation allows the host to access the PAC1000 as a memory-mapped peripheral.

Figure 6. Host Interface Architecture



Host Interface (Con't)

An example of FIFO usage is shown in Figure 7. When command or data information is available in the FIFO, the FIFO Output Ready (FIOR) interrupt (interrupt 5) triggers. If the FIOR interrupt is masked, then the FIOR status may be polled under program control. If HAD5 equals 1, the branch address location specified by MOVE is the Program Memory Address which contains the user specified instruction or sub-routine which executes the command. A JUMP or CALL FIFO control instruction performs a jump or call to the location specified by MOVE. If HAD5 equals 0, an RDFIFO instruction can transfer the FIFO contents into the register specified by HAD[4:0].

For further explanation, refer to the diagram below. Beginning at the location specified by MOVE, a user defined program exists which is going to load data into CPU registers 0,1,2, and 3 in four consecutive cycles from the next four FIFO locations. If one of the four FIFO locations contains a command (FICD=1), interrupt level 7 occurs (highest level). Loading a command into a CPU or other data register is not allowed. If this occurs, FIXP (FIFO exception) will be generated.

Following the execution of this routine, the Control Section is ready for its next instruction.

The FIFO drives three internal flags which can also be programmed to interrupt the PAC1000. They are:

- $\overline{\text{FIIR}}$ (FIFO full) and FIXP (FIFO exception), which drive INT7.
- FIOR (FIFO output ready), which drives INT5.

Table 2. Host Interface Functions

$\overline{CS}$	$\overline{RD}$	$\overline{WR}$	HAD5	HAD[4:0]	HD[15:0]	Function
0	1	0	0	Register Address	Data	Write data to FIFO
0	1	0	1	X	Command	Write command to FIFO
0	0	1	0	00100	X	Reset FIFO
0	0	1	0	00011	X	Reset status register
0	0	1	0	00010	Data	Read program counter
0	0	1	0	00001	Data	Read status register
0	0	1	0	00000	Data	Read data output register

Host Interface (Con't)

Data I/O Registers

Input and Output Data Registers are used to communicate with the Host Data (HD) bus. CPU Registers may be loaded directly from the Data Input Register (DIR) without passing through the FIFO. Similarly, the PAC1000 may be read via the Data Output Register (DOR).

Program Counter

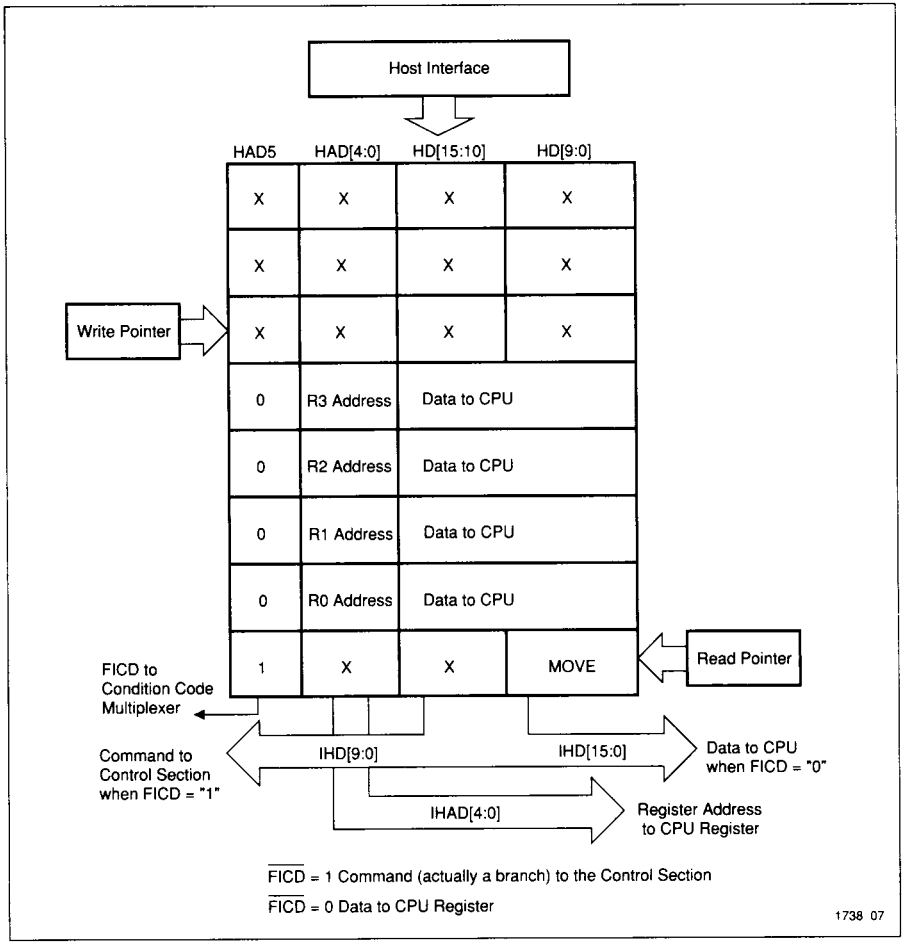
The Program Counter may be read via the Host Data bus. This allows a host to monitor the Program Memory address bus. It can also be used to drive external memory devices for expansion of the Control Port.

Status Register

The Status Register (SR), shown in Figure 8, monitors all internal status. Status bits can be set only by program execution. The SR can be read or cleared as specified in the Host Interface Functions table.

All SR flags are active high (1) and are latched at the rising edge of the clock.

Figure 7. Example of FIFO Block Diagram and Usage



4

Host Interface (Con't)

STAT11—(DBB) *Security Bit*, set when security is active:

1= Security active.

0= No security.

STAT10—*WSI Reserved.*

STAT9—(FIXP) *FIFO Exception*, set when the CPU receives a command or Control Section receives data:

1= Command or data received.

0= No exception occurred.

STAT8—(FIIR) *FIFO-Input Ready*, set when there is at least one vacant location in the FIFO:

1= FIFO ready for input.

0= FIFO not ready for input.

STAT7—(CY) *Carry Flag*, set when a carry (addition) or borrow (subtraction) occurs in CPU operations:

1= Carry occurred.

0= No carry occurred.

STAT6—(Z) *Zero Flag*, set when the result of a CPU operation is zero:

1= Zero occurred.

0= No zero occurred.

STAT5—(O) *Overflow Flag*, set when an overflow occurs during a two's complement operation:

1= Overflow occurred.

0= No overflow occurred.

STAT4—(S) *Sign Bit*, set when the most significant bit of the result of the previous CPU operation is negative:

1= Result is negative.

0= Result is positive.

STAT3—(STKF) *Stack Flag*, set when the stack is full:

1= Stack is full.

0= Stack is not full.

STAT2—(BRKPNT) *Breakpoint Flag*, set when the address in the breakpoint register is equal to the EPROM address:

1= Breakpoint occurred.

0= No breakpoint occurred.

STAT1—(BCZ) *Block Counter Zero*, set when the counter decrements to all 0s:

1= Block Counter reached zero.

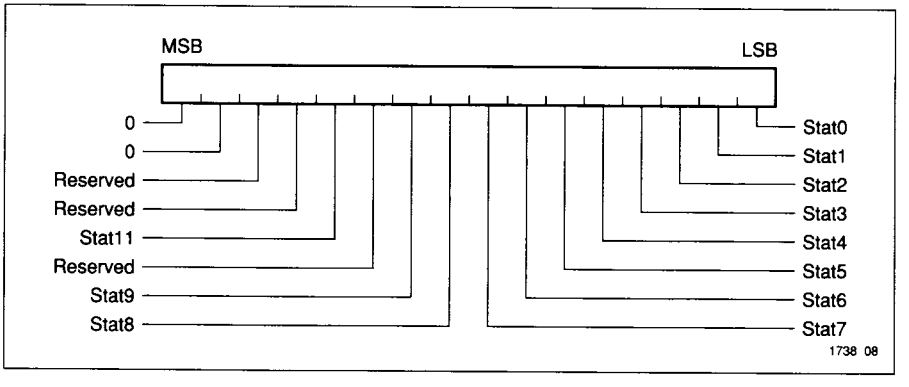
0= Block Counter is not zero.

STAT0—(ACO) *Address Counter Ones*, set when the counter increments to all 1s:

1= Address Counter reached all ones.

0= Address Counter is not all ones.

Figure 8. Status Register



Control Section

The control section, shown in Figure 9, consists of a number of blocks which are concerned with the sequencing of the control programs in the PAC1000. These are:

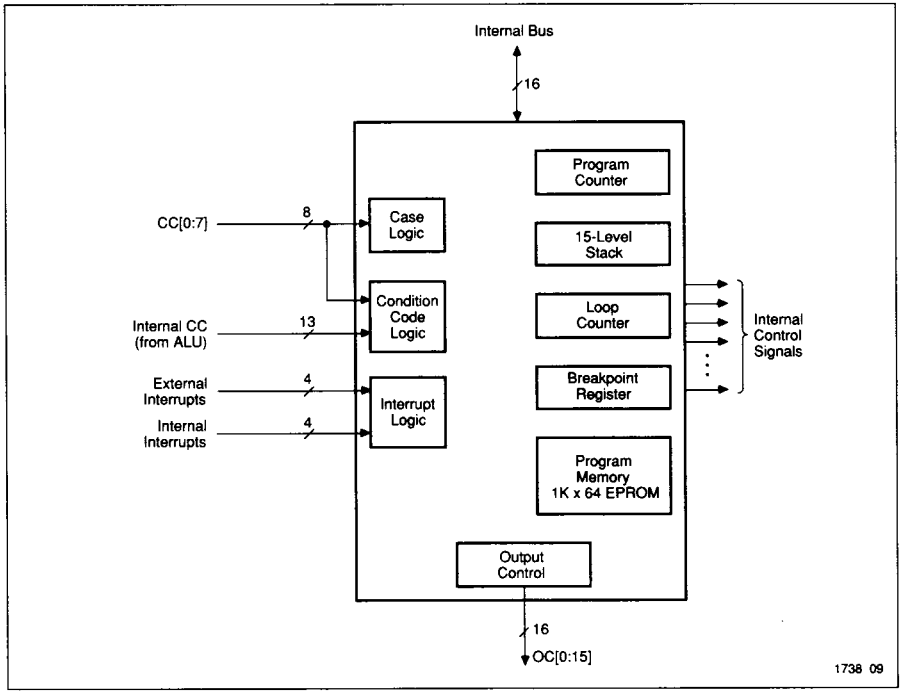
- Program Memory
- Security
- 15-Level Stack
- Program Counter
- Loop Counter
- Breakpoint Register
- Condition Codes
- Case Logic
- Interrupt Logic
- Output Control

Each block is described in detail below.

Parallel Operations

The PAC1000 can perform three simultaneous operations within a single instruction cycle, as shown in Figure 10. The ability to fetch an instruction from the Program Memory, execute it, and output a result within 50 nsec is due to a highly parallel structure.

Figure 9. Control Architecture



4

Control Section (Con't)

Program Memory

The Program Memory is a 1Kx64 high-speed EPROM. This on-board-memory allows the PAC1000 to operate in embedded control applications and eliminates the need for external memory components. Using an erasable memory allows program code to be modified for debug and/or field upgrades. The Program Memory is easily programmed using the WSI MagicPro™ (Memory and PSD Programmer).

Only sixteen Program Memory locations are reserved. The rest of the 1024 locations are available for applications.

Program memory is segmented as follows:

Address	Function
000H	Reset pointer program to here
000H–007H	User Defined Initialization Routine
008H–00FH	Interrupt Vector Locations
010H–3FFH	User-Defined Application Programs

Upon receiving a reset, the Program Counter is forced to address 000H. This location may contain a jump or call which branches to an initialization routine. Alternatively, the first eight locations of memory may be used as an initialization/configuration routine.

Security

User programs may be protected by setting a security bit during EPROM programming. Thereafter, the EPROM contents cannot be read externally. When the EPROM is erased, the security bit is cleared.

15-Level Stack

The 15-level Stack stores the return address following subroutine calls, interrupt service routines and the contents of the Loop Counter inside nested loops. When the stack is full, the STKF condition becomes true, and an interrupt (INT7) will occur. The interrupt service routine will overwrite the top of the stack.

Popping from an empty stack produces the previous top of stack value; pushing on a full stack overwrites the top of the stack.

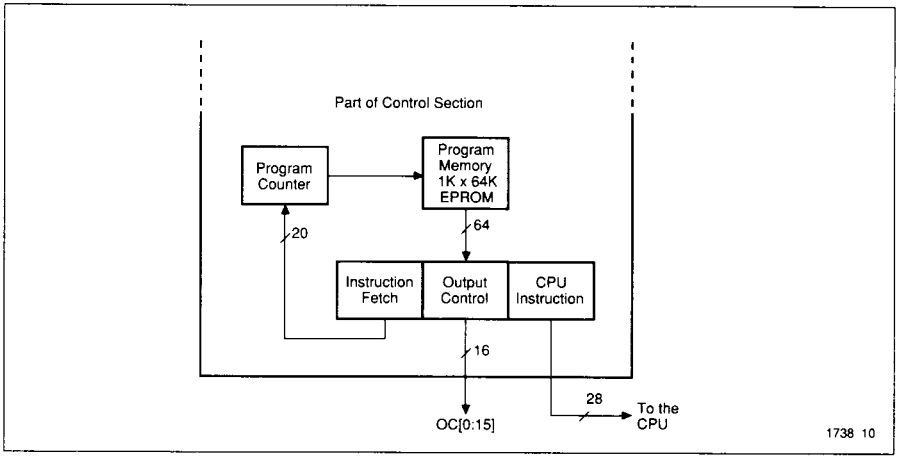
Program Counter

The 10-bit Program Counter (PC) generates sequential addressing to the 1K word Program Memory. Upon reset the PC is loaded with a 000H. From this point the value of the Program Counter is determined by program execution or interrupts.

Any JUMP or Case instruction that is executed loads the Program Counter with the destination address. CALL instructions or interrupts cause PC + 1 to be pushed onto the stack. The RETURN instruction loads the Program Counter from the stack with the value of the return address. This value may have previously been placed on the stack by a CALL or interrupt.

The PC can also be loaded from the Command/Data FIFO causing program execution to commence at an address provided by the host.

Figure 10. Parallel Operations



Control Section (Con't)

Loop Counter

The Loop Counter (LC) has two functions:

- 10-bit down counter that supports the LOOP instruction.
- Branch Register that can be loaded from the CPU Register File or Program Memory and used as an additional source of branching to Program Memory.

The LC can be loaded with values up to 1023. Loop initialization code places a value in LC. Loop termination code tests the counter for a zero value and then decrements LC. The loop count can be a constant, or it can be computed at execution time and loaded into LC from the CPU. The LC register can also be used as a CALL or JUMP execution vector. The content of the LC is automatically saved on (or retrieved from) the Stack when the program enters (or leaves) a nested loop.

A loop count will be loaded into the LC when a FOR instruction is encountered. This count can be a fixed value or it can be calculated and loaded from the CPU. The ENDFOR instruction will test the Loop Counter for a zero value. If this condition is not met, then the LC will be decremented by one. The program loop will continue until the count value equals zero. In a nested loop, the FOR instruction will load a new value to the LC and push the previous value to the stack.

Debug Capabilities

The PAC1000 provides breakpoint and single step capabilities for debugging application programs.

Breakpoint Register

The Breakpoint Register (BR) is a 10-bit register used for real time debug of the PAC1000 application program.

The Breakpoint Register can be loaded from one of two sources, either a constant value specified in the Program Memory or a calculated value loaded from the CPU. When the Program Memory address matches the contents of the Breakpoint Register an interrupt (INT 6) occurs. A service routine should exist in Program Memory which then performs the required procedure.

Single Step

Single step is a debugging mode in which the currently-executing program is interrupted by interrupt 6 after the execution of every instruction. The interrupt 6 service routine should reside in Program Memory.

Bit 8 in the Mask Register determines whether the PAC1000 is in a breakpoint mode (mask-bit 8 equals 0) or in a single step mode (mask-bit 8 equals 1).

Both breakpoint and single step use interrupt 6. The interrupt 6 service routine will typically dump the contents of the PAC1000 internal registers into external SRAM devices for examination by the user.

Condition Codes

The Condition Code (CC) logic operates on 21 individual program test conditions. Each condition can be tested for true or not true. The PAC1000 can also test up to four conditions simultaneously. For this feature refer to the section titled *Case Logic*.

Control Section (Con't)

User-Specified Conditions

User-Specified Conditions are treated in the same manner as internally generated test conditions. CC0—CC7 should be connected directly to the corresponding PAC1000 input pins. These signals must satisfy the required setup time to be serviced in the next cycle.

CPU Flags

CPU flags are internally generated. They reflect the status of the previous CPU arithmetic operation. These signals are internally latched and are valid only for one instruction (the instruction following their generation). The flags for arithmetic operations are defined as follows:

Zero (Z)—The result of the previous CPU operation is zero (Z=1).

Carry (CY)—The result of the previous CPU operation generated a carry (addition) or borrow (subtraction) (CY=1).

Overflow (O)—The previous two's complement CPU operation generated an overflow (O=1).

Sign (S)—The most significant bit of the result of the previous CPU operation is negative (S=1).

FIFO Flags

FIFO flags allow the user to synchronize and monitor the operations that are performed on the FIFO by the host or by user's program.

Upon reset the FIFO flags are cleared, signifying an empty state. The meaning of the flags are as follows:

FIFO Output Ready (FIOR)—There is at least one word in the FIFO (FIOR=1).

FIFO Input Ready (FIIR)—FIFO is not full (FIIR=1). This flag can also be connected to the host through I/07.

FIFO Command/Data (FICD)—This flag indicates if the contents of the FIFO is a command or a data. This flag is generated directly from HAD5 (FICD=1 command, FICD=0 data).

FIFO Exception (FIXP)—This flag indicates that one of two events occurred: (a) FIFO data has been read as a command, or (b) a command has been read as data.

Stack-Full Flag

STACK FULL flag (STKF=1) indicates that the stack is 15 levels full. This condition will also generate an interrupt (INT7) if not masked.

Interrupt Flag

INTERRUPT flag (INTR =1) indicates that there is a masked interrupt pending. This flag is cleared when the interrupt is cleared.

Data Register Read Flag

DATA REGISTER READ flag (DOR) is a handshake flag between the host and the PAC1000, accessible only to the PAC1000. The flag is reset (DOR=0) when the PAC1000 writes into the Data Output Register. The flag is set (DOR=1) after the host has performed a read on the Data Output Register.

Counter Flag

Counter flags reflect the status of their respective counters. The PAC1000 utilizes two counters; the Address (A) counter is a 16/22-bit auto-incrementing up counter; the

Table 3. Condition-Code Logic

Test Group	Source	Conditions and Flags
User-Specified	External	CC0–CC7
CPU	Internal	Carry (CY), Zero (Z), Overflow (O), Sign (S)
FIFO	Internal	FIFO Command/Data (FICD), FIFO Output Ready (FIOR), FIFO Input Ready (FIIR), FIFO Exception (FIXP)
Counters	Internal	Address Counter Ones (ACO), Block Counter Zero (BCZ)
Stack	Internal	Stack Full (STKF)
Interrupt	External/Internal	Interrupt (INTR) is pending
Data register read	Internal	Data Output Register(DOR) has been read

Control Section (Con't)

Block (B) counter is an auto-decrementing 16-bit down counter. The counters' clock input signal is the same as the PAC1000's clock signal. Each counter can be individually enabled or disabled. When disabled, the output retains the last count. The counter flags are defined as follows:

ACO—*A Counter Ones*, set when the A counter has reached the value FFFFH, in the 16-bit mode, or the value 3FFFFFH in the 22-bit mode.

BCZ—*B Counter Zero*, set when the B counter has reached the value 0000H.

Case Logic

THE PAC1000 hardware implements two basic types of Case instructions: Case and Priority Case.

Case Instructions

Case instructions operate on any one of four different Case groups. Each Case group consists of a combination of four test conditions which can be tested in a single cycle. In that same cycle the PAC1000 will branch to one of the addresses contained in the sixteen memory locations following the instruction, depending on the status of the four inputs to the Case group being tested.

There are four Case Groups (sets of Case Conditions):

Case Group 0 (CG0): CC0–CC3.

Case Group 1 (CG1): CC4–CC7.

Case Group 2 (CG2):

- Z—Zero
- O—Overflow
- S—Sign
- CY—Carry

Case Group 3 (CG3):

- INTR—Interrupt
- BCZ—B Counter Zero
- FIOR—FIFO output Ready
- FICD—FIFO Command/Data

(The FIXP, ACO, STKF, FIIR, and DOR condition codes do not fall into any of the four Case groups.)

Priority Case Instructions

Priority Case instructions operate on the four internal and the four external interrupt inputs. In this mode of operation, interrupts are treated as prioritized test conditions and the priority encoder is used to generate a branch to the highest priority condition. The branch address is located in one of the nine memory locations following the Priority Case instruction. Priorities in this mode of operation are the same as in the Interrupt mode of operation. Once a Priority Case instruction is executed, the occurrence of a higher priority condition will not affect program execution until another Priority Case instruction is executed. For a Priority Case instruction to be executed, MODE0 of the Mask Register must be equal to zero (MODE0=0).

Interrupt Logic

The Interrupt Logic accepts eight inputs, four of them are generated externally and four are dedicated for internal conditions. The four external, user defined, inputs (INT0–INT3) are connected to pins INT0, INT1, INT2, and INT3. These are positive, rising-edge-triggered signals that have a maximum latency of two cycles. Each interrupt has a reserved area in memory that should contain a branch to an interrupt service routine.

4

Table 4. Interrupt Assignments

Interrupt	Priority	Effect	Trigger Condition	Reserved Address
INT7	Highest	Internal	FIXP+ACO+STKF+$\overline{\text{FIIR}}$	00FH
INT6		Internal	BRKPT	00EH
INT5		Internal	FIOR	00DH
INT4		Internal	Software Interrupt (SWI)	00CH
INT3		External	INT3	00BH
INT2		External	INT2	00AH
INT1		External	INT1	009H
INT0	Lowest	External	INT0	008H

Control Section (Con't)

Clearing a serviced interrupt is performed automatically. When the interrupt is serviced, the internally generated vector is decoded to clear the serviced interrupt. In addition, the user can clear any pending interrupt by using the Clear Interrupt Instruction (CLI).

Interrupt Mask Register

The Interrupt Mask Register, shown in Figure 11, allows individual interrupts to be masked. Setting a Mask Register bit to a 1 masks the associated interrupt. To unmask an interrupt, the appropriate Mask Register bit must be reset to 0.

When the PAC1000 is reset,the Mask Register will mask all interrupts and the Mode Register will select the non-interrupt mode. To select the interrupt mode the MODE0 bit (see Configuration Register section in this document) should be set to 1 (MODE0=1).

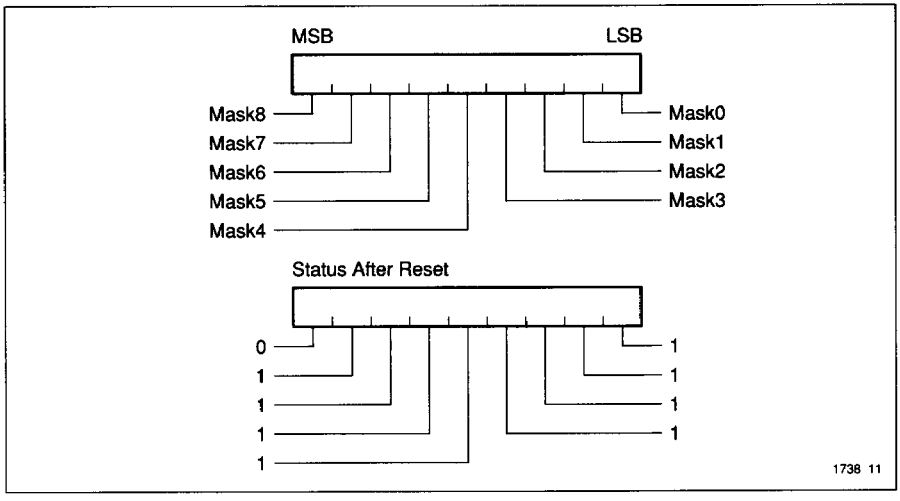
Mask8 is used to select INT6 to be either a single-step interrupt (when Mask8=1) or a breakpoint interrupt (when Mask8=0) .See the section on Debug Capabilities for further details.

Table 5. Interrupt Definitions

Interrupt	Triggered By
INT7[1]	FIFO Exception (FIXP)
	Address Counter contains all Ones (ACO)
	Stack Full (STKF)
	FIFO Full (Not FIFO Input Ready, $\overline{FIIR}$)
INT6[2]	Breakpoint or Single Step occurrence
INT5	FIFO Output Ready (FIOR)
INT4	Always pending; triggers when unmasked by program execution
INT3	User-defined
INT2	User-defined
INT1	User-defined
INT0	User-defined

Notes:
1. The INT7 interrupt handler checks the source of the interrupt by testing the condition code.
2. See Interrupt Mask Register, Mask8.

Figure 11. Interrupt Mask Register



Control Section (Con't)

Output Control

The Output Control bus (OUTCNTL) consists of 16 latched Output Control signals. These signals can be changed on a clock to clock basis. For every Program Memory location there is a dedicated field which specifies the value of the Output Control bus. The OUTCNTL Operation places this value on the Output Control bus. The OUTCNTL Operation can be performed in parallel with any other PAC1000 instructions.

The OUTCNTL bus can be used to control external events on a clock to clock basis.

Counters

The PAC1000 contains a 16 or 22-bit Address Counter and a 16-bit Block Counter. Each of these counters can change count on a clock to clock basis or can be internally or externally enabled or disabled on a clock to clock basis. These counters are in addition to the Loop and Program Counters of the Control Section.

Address Counter

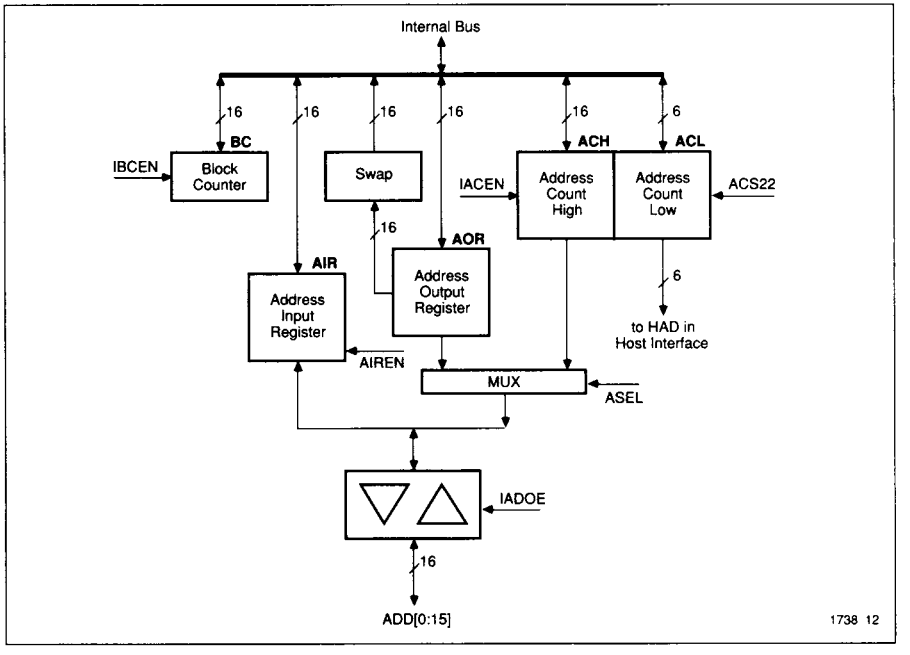
The Address Counter (AC), shown in Figure 12, is a 16- or 22-bit ascending counter that can be loaded or read by the CPU and enabled/disabled with the ACEN bit of the Control Register. (This control is also available externally through the I/01 pin; see I/O and Special Functions). While enabled, the counter will increment by one every rising edge of the clock.

The ACO flag indicates that the value of the counter is all ones. This flag stays latched until the counter is loaded with a new value. The counter will continue to count until disabled. ACO is a condition code and a member of a Case Group; see the Control Section description for more details. ACO can also generate an internal interrupt 7, if enabled.

In the 16-bit mode, the counter outputs (ACH) are available through the ADD bus. The count is gated to the ADD bus by setting the ASEL bit (CTRL9) of the Control Register.

In the 22-bit mode, the higher 16 bits (ACH) are available through the ADD bus and the six low order bits (ACL) are available through the Host Address (HAD) bus. These low order bits are multiplexed with the host address lines. The address lines from the host which drives the HAD bus must be placed in the high impedance state before the lower 6-bits (ACL) of the Address Counter can be read.

Figure 12. Address and Block Counter



Counters (Con't)

Selecting the 16- or 22-bit count mode is performed by setting or resetting the ACS22 bit in the I/O Configuration Register.

The address Output Register is an alternate source of address outputs; it is selected by resetting the ASEL bit of the Control Register. In this mode the CPU can be used to provide address generation and the Address Counter can be used as an event counter.

Block Counter

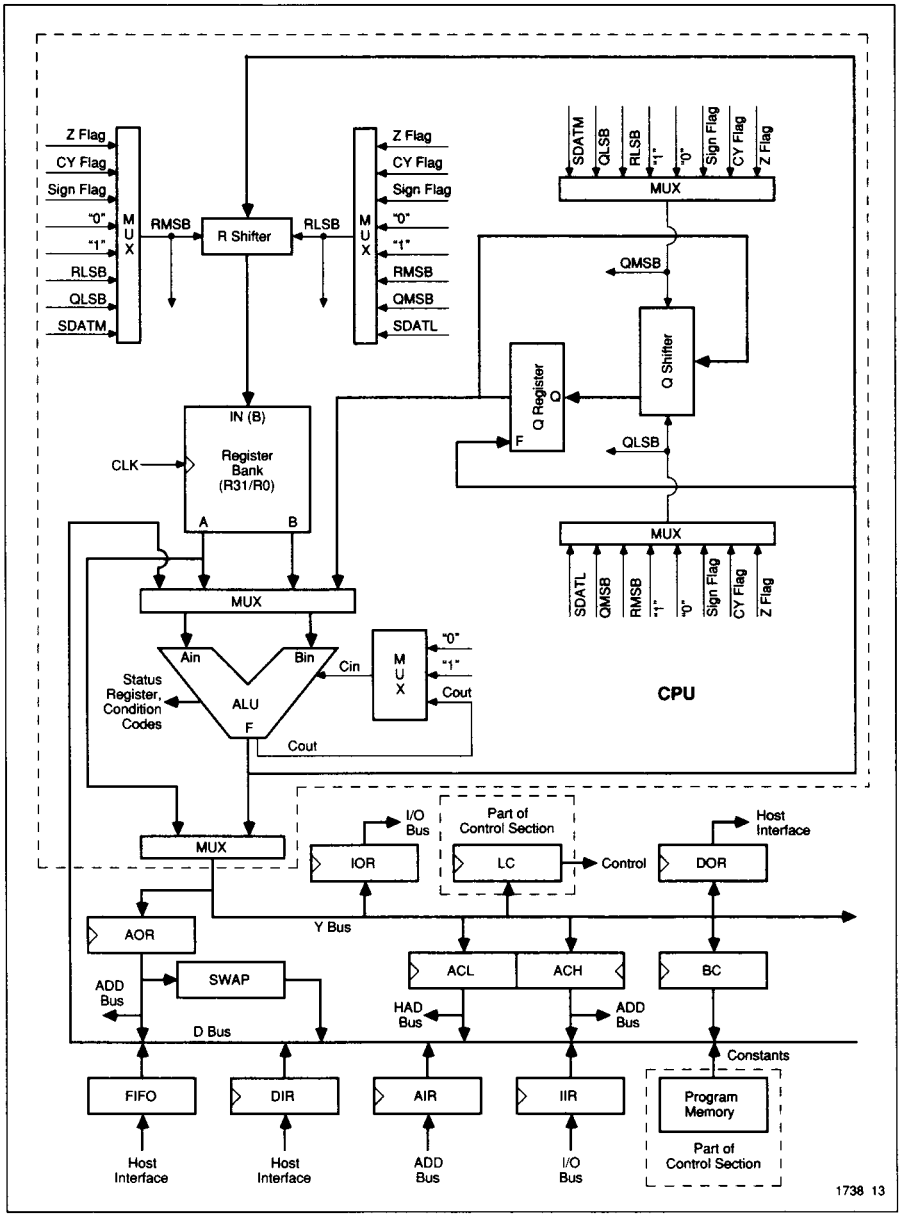
The Block Counter (BC) is a 16-bit down counter. It is enabled by the BCEN bit of the Control Register. It is useful as a counter for DMA transfers. The BCEN signal is (optionally) available externally through the I/O0 bit (see I/O and Special Functions). While enabled, the counter will decrement by one every rising edge of the clock. The BCZ flag indicates that the counter reached the zero value. After the occurrence of an all 0s condition the Block Counter will continue down counting until disabled. The flag is latched and can be cleared by loading a new value into the Block Counter. BCZ is a condition code and a member of a Case Group; see the Control Section description for more details.

Both counters may be read without disabling the count operation and loaded via the CPU.

Central Processing Unit

The CPU, shown in Figure 13, performs 16-bit operations in a single clock cycle. It contains 33 general purpose registers (R0...R31, and Q). The Q register can be used in conjunction with any of the R0...R31 registers to perform double precision shift operations. The main building blocks are the register bank (R0...R31), Q register, ALU, Y-bus devices, and D-bus devices. The register bank supplies up to two 16-bit registers, one of which is always the destination register.

Figure 13. CPU Block Diagram



Central Processing Unit (Con't)

The ALU operates on up to two external operands that are selected by its input MUX. In every instruction, 1 of the 10 D-bus devices (AOR, SWAP, ACL, ACH, BC, FIFO, DIR, AIR, IIR, and Program Store) or a member of the register bank or the Q register outputs, can be selected as an operand source to the ALU. The possibilities are shown in Figure 14. During ALU operations, three options can be selected to provide the carry-in (Cin) input: 0, 1, or the previous latched carry-out (adequate for multiple precision operations).

The ALU's output or a selected register can be loaded into one of the seven Y-bus devices (IOR, AOR, LC, DOR, ACL, ACH, or BC) every instruction cycle. This can happen in parallel with the feedback path from the ALU's output that is directed either to the Q register or to the destination register of the register bank.

Figure 14. CPU Sources and Destinations

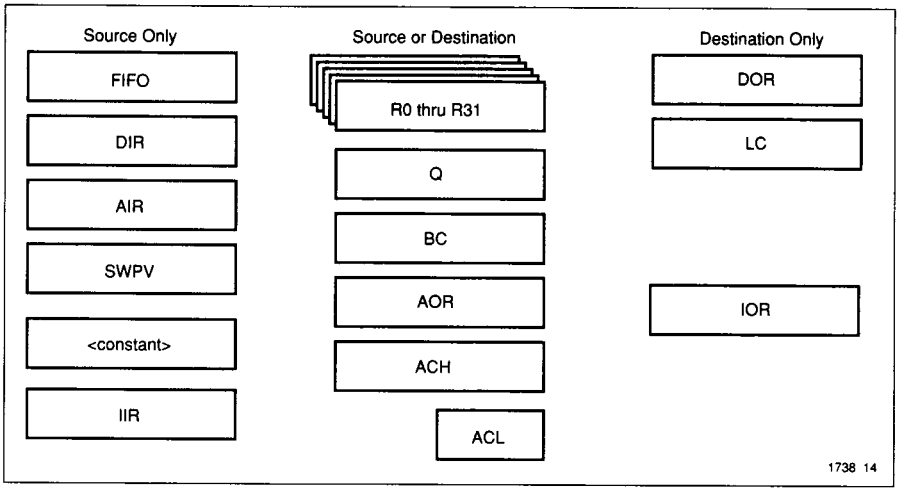


Table 6. CPU Operand Mnemonics

Mnemonic	Description
ACH or ACH/ACL	16- or 22-bit Auto-incrementing Counter, or General Purpose Registers
AIR	Address Input Register
AOR	Address Output Register
BC	Block Counter (16-bit auto-decrementing), or General Purpose Register
<constant>	Constant values in Program Storage
DIR	Data Input Register
DOR	Data Output Register
FIFO	Input Data from FIFO
IIR	I/O Input Register
IOR	I/O Output Register
LC	Program Loop Counter
Q	16-bit CPU Register
R0–R31	16-bit CPU Registers
SWPV	Byte Swap version of AOR

Central Processing Unit (Con't)

CPU operations can be performed on one, two or three operands. Each operation is performed in a single clock cycle. In two- or three-operand instructions, one of the operands must be a CPU internal register (R0...R31, or Q).

CPU operations are performed independently of operations in the counters, Host Interface, Output Control, and Program Control.

Arithmetic Operations

The CPU can perform the following arithmetic operations:

- Addition
- Subtraction
- Increment
- Decrement
- Compare

Logic Operations

The CPU can perform the following logic operations:

- AND
- OR
- Invert
- Exclusive OR
- Exclusive NOR

Shift Operations

Single shift operations, shown in Figure 15, can occur either to the left or to the right, with or without the Q register. Shift instructions specify the sources that are shifted into the corresponding registers.

All shift operations can be executed in the same clock cycle as an arithmetic or logic operation. The arithmetic or logic operation is executed first; the result is shifted and then stored in the register file. The shift can be either left or right.

The CPU can perform the following shift operations:

- Single-precision, left or right, within a general-purpose register (R0...R31, or Q).
- Double-precision, left or right, between an R0...R31 register and the Q register.

The LSB and MSB of the general-purpose registers are each fed by an eight-to-one multiplexer.

The sources and destinations for shift operation are given below:

Shift Right

Zero Flag (Z)

Carry Flag (CY)

Sign Flag (S)

Binary 0 (0)

Binary 1 (1)

Least-significant bit of this register (RLSB)

Least-significant bit of the Q register (QLSB)

Serial I/O port (SDATM)

Shift Left

Zero Flag (Z)

Carry Flag (CY)

Sign Flag (S)

Binary 0 (0)

Binary 1 (1)

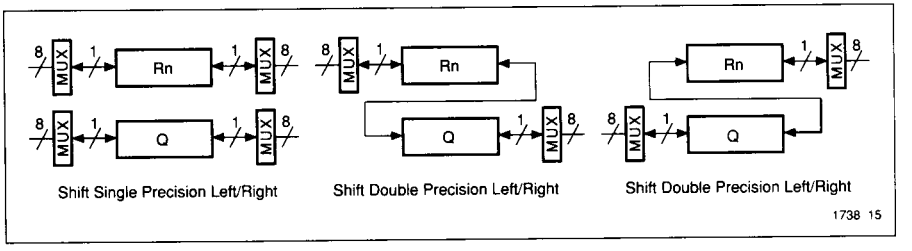
Most-significant bit of this register (RMSB)

Most-significant bit of the Q register (QMSB)

Serial I/O port (SDATL)

4

Figure 15. Shift Operations



Central Processing Unit (Con't)

Rotate Operations

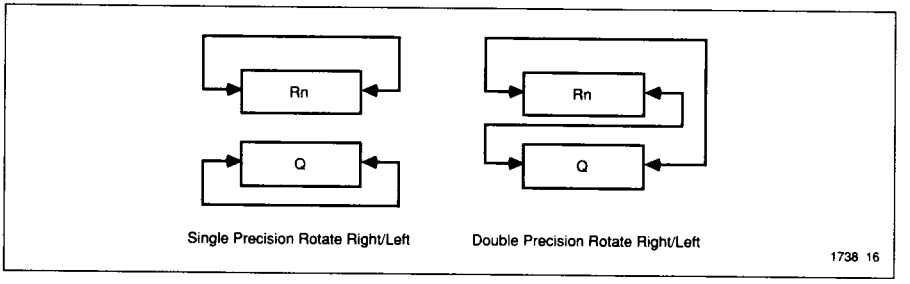
The CPU can perform the following rotate operations, as shown in Figure 16:

- Single-precision, left or right, within a general-purpose register (R0...R31, or Q).
- Double-precision, left or right, between an R0...R31 register and the Q register.

Multiple Precision Operations

The carry-out in each instruction can be used in the next instruction for multiple precision operations (e.g., ADDC). This feature enables the user to implement complex arithmetic operations such as division or multiplication in several clock cycles.

Figure 16. Rotate Operations

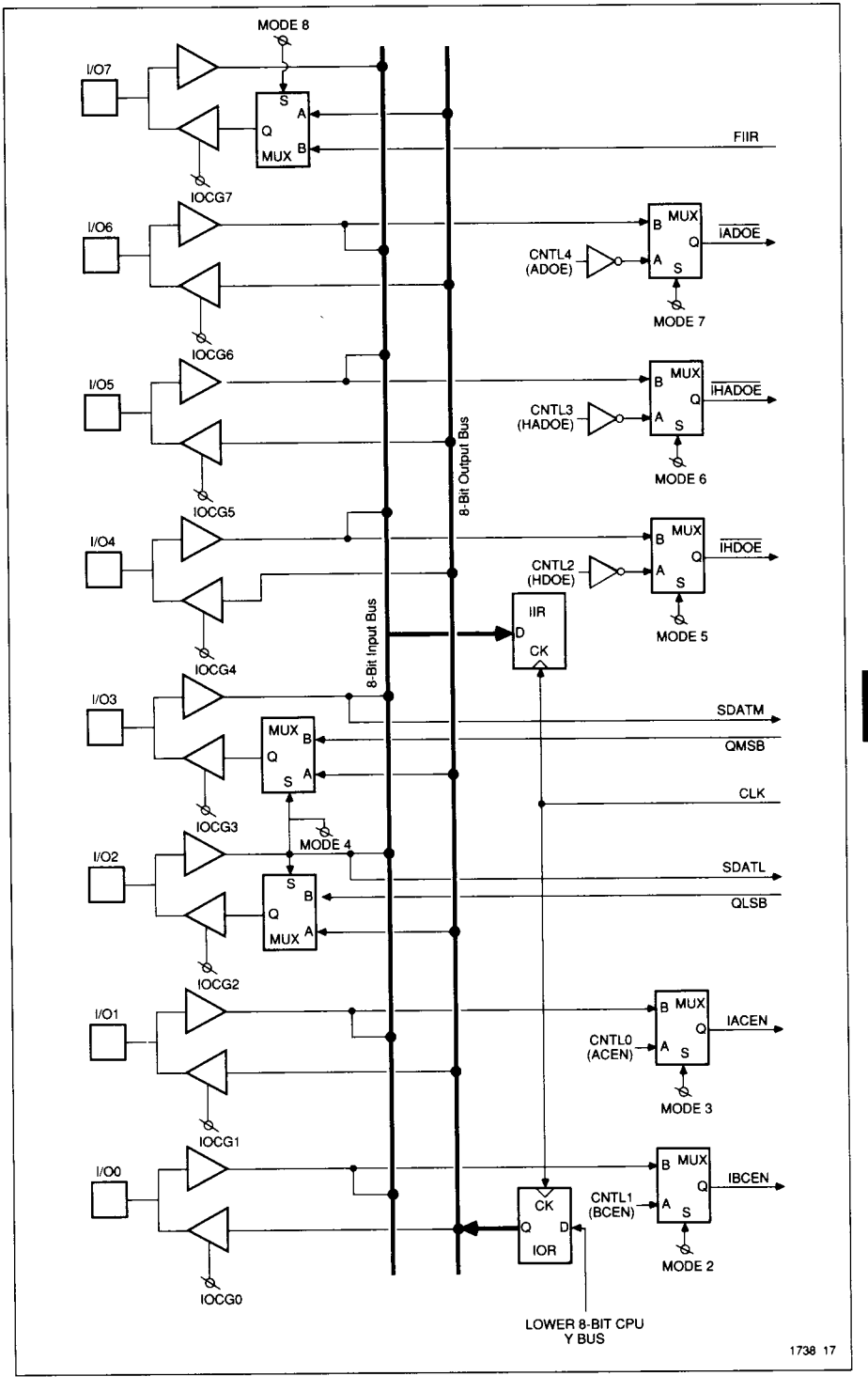


I/O and Special Functions

The I/O bus, shown in Figure 17, consists of eight lines which can be individually programmed as inputs or outputs. These lines can also be programmed to perform Special Functions. The functions of these pins are defined by the Mode Register and I/O Configuration Register (see Configuration Register Section). The I/O and Special Functions map according to the table. The I/O lines must first be configured as inputs or outputs via the I/O Configuration Register; the Special Function option can then be enabled via the Mode Register. Individual special function control is shown in the accompanying table.

Once a Special Function has been enabled, the corresponding internal control function is automatically disabled. Conversely, when a Special Function is disabled, control of the corresponding internal control function is returned to the Control Register (see Configuration Register). Because the Inputs in the I/O Register are clocked on each cycle, the status of the special function can also be read to the CPU.

Figure 17.
I/O and Special Function Bus



Configuration Registers

The Configuration Registers allow the user to control and configure different operating modes of the PAC1000. The three 10-bit Configuration Registers are the Control Register, I/O Configuration Register, and Mode Register. Each register has an associated instruction which allows individual register bits to be modified.

Control Register

The Control Register, shown in Figure 18, provides for internal control of key functions within the PAC1000 . Several of these functions can alternatively be controlled externally through the I/O bus (see I/O and Special Functions). The Control Register is modified on the falling edge of the clock.

Table 7. I/O Pins and Special Functions

Pin	Special Function	Direction	Description
I/O7	FIIR	output	FIFO Input Ready. FIFO not full.
I/O6	ADOE	input	Address Output Enable
I/O5	HADOE	input	Host Address Output Enable
I/O4	HDOE	input	Host Data Output Enable
I/O3	QMSB	bidirectional	Q Register MSB
I/O2	QLSB	bidirectional	Q Register LSB
I/O1	ACEN	input	Address Counter Enable
I/O0	BCEN	input	Block Counter Enable

Table 8. Special-Function Control

Special Function	Pin Name	I/O Configuration	Mode
FIIR	I/O7	IOCG7=1 (output)	MODE8=1
ADOE	I/O6	IOCG6=0 (input)	MODE7=1
HADOE	I/O5	IOCG5=0 (input)	MODE6=1
HDOE	I/O4	IOCG4=0 (input)	MODE5=1
QMSB	I/O3	IOCG3=1 (output)	
		IOCG3=0 (input)	MODE4=1
QLSB	I/O2	IOCG2=1 (output)	
		IOCG2=0 (input)	MODE4=1
ACEN	I/O1	IOCG1=0 (input)	MODE3 =1
BCEN	I/O0	IOCG0=0 (input)	MODE2 =1

Configuration Registers (Con't)

ASEL (CTRL9)—*Address Select.* Selects the source that will write to the Address bus:

1= Address Counter.

0= Address Output Register (AOR).

AIREN (CTRL8)—*Address Input Register Enable.* Enables and disables writing to the Address Input Register from the ADD Port:

1= Enable writing to Address Input Register (AIR).

0= Disable writing to Address Input Register (AIR).

DIREN (CTRL7)—*Data Input Register Enable.* Enables and disables writing to the Data Input Register (DIR) from the HD Port:

1= Enable writing to Data Input Register (DIR).

0= Disable writing to Data Input Register (DIR).

HDSEL1 (CTRL6) and HDSEL0 (CTRL5)—*Host Data Select.* Select the source to be connected to Host Data (HD) bus:

HDSEL1 (CTRL6)	*HDSEL0 (CTRL5)*	*Selection*
0	0	FIFO—Peripheral Mode
0	1	Data Output Register
1	0	Status Register
1	1	Program Counter

ADOE (CTRL4)—*Address Output Enable.* Selects direction of Address bus (ADD) for next clock cycle:

1= Output (see ASEL).

0= Input (see AIREN).

HADOE (CTRL3)—*Host Address Output Enable.* Selects direction of Host Address (HAD) bus for next clock cycle:

1= Output (driven from ACL Register).

0= Input (into the FIFO).

HDOE (CTRL2)—*Host Data Output Enable.* Selects Direction of Host Data (HD) bus for next clock cycle:

1= Output (See HDSEL0 and HDSEL1).

0= Input (See DIREN).

BCEN (CTRL1)—*Block Counter Enable.* Enables and disables Block Counter:

1= Enable Counting on next rising clock edge.

0= Disable Counting on next rising edge.

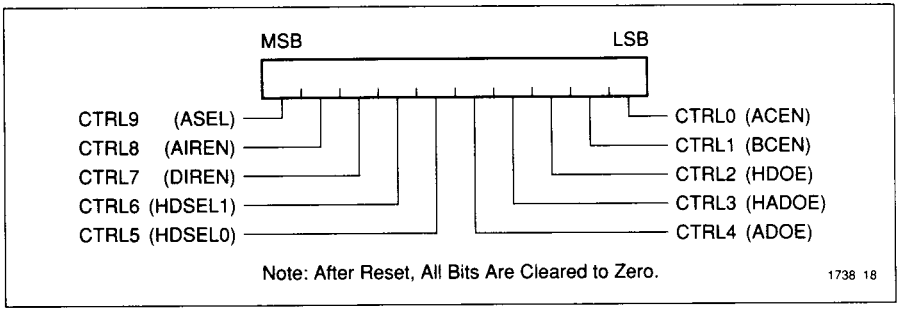
ACEN (CTRL0)—*Address Counter Enable.* Enables and disables Address Counter:

1= Enable Counting on next rising clock edge.

0= Disable Counting on next rising clock edge.

4

Figure 18. Control Register



Note: After Reset, All Bits Are Cleared to Zero.

1738 18

Configuration Registers (Con't)

I/O Configuration Register

The I/O Configuration Register, shown in Figure 19, controls the direction of the individual lines of the I/O bus as well as configuring the Address Counter. Each I/O pin can be configured independently to be a general purpose input or output, or each can serve a special function (see I/O and Special Function). The I/O Configuration Register is also used to configure the Address Counter as a 16-bit counter with a maximum count of FFFFH or as a 22-bit counter with a maximum count of 3FFFFFH. The I/O Configuration Register is modified on the falling edge of the clock.

ACS22 (IOCG9)—Configures Address Counter as a 22- or 16-bit counter:

1= 22-bit counter.

0= 16-bit counter.

I/O7 (IOCG7)—Selects direction of I/O7 pin:

1= Output.

0= Input.

I/O6 (IOCG6)—Selects direction of I/O6 pin:

1= Output.

0= Input.

I/O5 (IOCG5)—Selects direction of I/O5 pin:

1= Output.

0= Input.

I/O4 (IOCG4)—Selects direction of I/O4 pin:

1= Output.

0= Input.

I/O3 (IOCG3)—Selects direction of I/O3 pin:

1= Output.

0= Input.

I/O2 (IOCG2)—Selects direction of I/O2 pin:

1= Output.

0= Input.

I/O1 (IOCG1)—Selects direction of I/O1 pin:

1= Output.

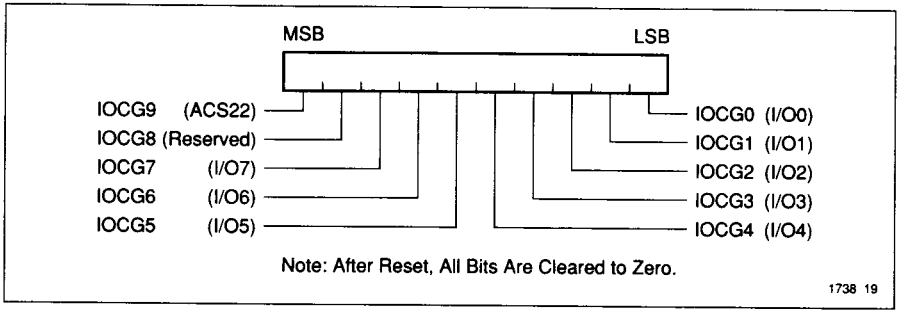
0= Input.

I/O0 (IOCG0)—Selects direction of I/O0 pin:

1= Output.

0= Input.

Figure 19. I/O Configuration Register



Note: After Reset, All Bits Are Cleared to Zero.

1738 19

Configuration Registers (Con't)

Mode Register

The Mode Register, shown in Figure 20, allows the user to externally control and monitor key elements within the PAC1000 which would (alternatively) be controlled internally through the Control Register. Enabling a Special Function in the Mode Register disables the corresponding function in the Control Register. The Special Function input pins are shared with the general purpose I/O pins. The direction of the appropriate pin must be set in the I/O Configuration Register prior to programming the Mode Register.

The Mode Register can also be used to reset the FIFO as well as program the interrupt controller to generate either interrupts or Priority Test Conditions. See the discussion on "Priority Case" in the *Condition Code* section, above.

After Reset, all Mode Register bits equal zero. The Mode Register is modified on the falling edge of the clock.

The use of the Mode Register and I/O Configuration register for Special Functions is shown in the Special Function Settings table.

FIRST (MODE9)—*FIFO Reset.* (If held high, FIFO cannot receive information):

1= Initiate FIFO Reset (FIRST).

0= Complete FIFO Reset (FINRST).

FIIR (MODE8)—*FIFO Input Ready:*

1= I/O7 becomes output for the FIFO Input Ready (FIIR) flag.

0= I/O7 becomes general purpose I/O (IO7).

ADOE (MODE7)—*Address Output Enable:*

1= I/O6 becomes input for the Address Output Enable (AOE).

0= I/O6 becomes general purpose I/O (IO6).

HADOE (MODE6)—*Host Address Output Enable:*

1= I/O5 becomes input for Host Address Output Enable (HADOE).

0= I/O5 becomes general purpose I/O (IO6).

HDOE (MODE5)—*Host Data Output Enable:*

1= I/O4 becomes input for Host Data bus Output Enable HDOE).

0= I/O4 becomes general purpose I/O (IO4).

SIOEN (MODE4)—*Serial I/O Enable:*

1= I/O3 and I/O2 become MSB and LSB (respectively) of the CPU's Q register (SIO).

0= I/O3 and I/O2 become general purpose I/O ACEN(MODE3).

ACEN (MODE3)—*Address Counter Enable:*

1= I/O1 becomes input for Address Counter Enable (ACEN).

0= I/O1 becomes general purpose I/O.

BCEN (MODE2)—*Block Counter Enable:*

1= I/O0 becomes input for Block Counter Enable (BCEN).

0= I/O0 becomes general purpose I/O.

Reserved (MODE1)

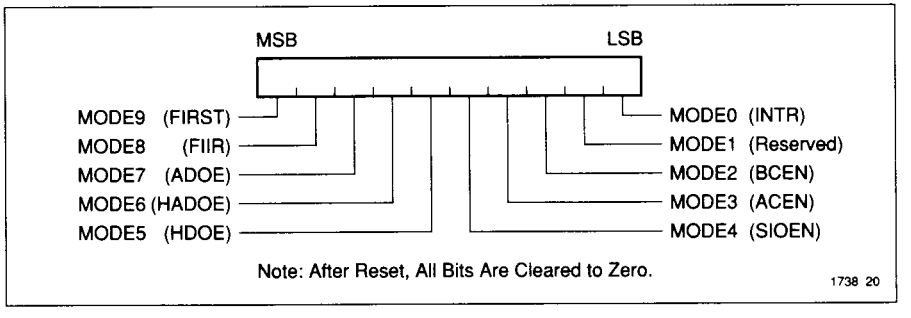
INTR (MODE0)—*Interrupt/Priority-Case Mode:*

1= Select Interrupt mode (INTR).

0= Selects Priority Case mode (PCC).

4

Figure 20. Mode Register



Note: After Reset, All Bits Are Cleared to Zero.

1738 20

State Following Reset

Whenever the PAC1000 $\overline{\text{RESET}}$ input is driven low for at least two processor clocks, the chip goes through reset. The next two tables describe the PAC1000 signal and internal register states following reset.

Table 9. Special Function Settings

Mode Bit	I/O Configuration Bit	Function
MODE8=1	IOCG7=1	FIIR flag output on I/O7
MODE7=1	IOCG6=0	$\overline{\text{ADOE}}$ provided by I/O6
MODE6=1	IOCG5=0	$\overline{\text{HADOE}}$ provided by I/O5
MODE5=1	IOCG4=0	$\overline{\text{HDOE}}$ provided by I/O4
MODE4=1	IOCG3=1	MSB of Q register output on I/O3
MODE4=1	IOCG3=0	I/O3 can be shifted into MSB of Q register or destination register
MODE4=1	IOCG2=1	LSB of Q register output on I/O2
MODE4=1	IOCG2=0	I/O2 can be shifted into LSB of Q register or destination register
MODE3=1	IOCG1=0	ACEN provided by I/O1
MODE2=1	IOCG0=0	BCEN provided by I/O0

Table 10. Signal States Following Reset

Signal	Condition
HAD[5:0]	Input
HD[15:0]	Input
IO[7:0]	Input
ADD[15:0]	Input
OC[15:0]	0000H

Table 11. Internal States Following Reset

Component	Contents
ACH Register	0
ACL Register	0
AOR Register	0
AIR Register	0
DOR Register	0
DIR Register	0
IOR Register	0
IIR Register	0
STATUS Register	0
I/O Configuration Register	0
CONTROL Register	0
Breakpoint Register	0
Mode Register	0
PC Register (Program Counter)	0
MASK Register	011111111B
BC Register	FFFFH
R31–R0 Registers	Unknown
Q Register	Unknown
LC Register	Unknown
FIFO Locations	Unknown
FIFO Flags	Empty

Electrical and Timing Specifications

Table 12. Absolute Maximum Ratings

Storage Temperature	–65°C to +150°C
Voltage to any pin with respect to GND	–0.6V to +7V
V_{PP} with respect to GND	–0.6 V to +14.0V
ESD Protection	>2000V

Stresses above those listed here may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods of time may affect device reliability.

Table 13. Operating Range

Range	Temperature	V_{CC}
Commercial	0°C to +70°C	+5V ± 5%
Industrial	–40°C to +85°C	+5V ± 10%
Military	–55°C to +125°C	+5V ± 10%

Table 14. DC Characteristics

Over operating range with $V_{PP}=V_{CC}$

Parameter	Symbol	Test Conditions	Min	Max	Units
Output Low Voltage	V_{OL}	I_{OL}=8 mA		0.4	V
Output High Voltage	V_{OH}	I_{OH}=–4 mA	2.4		V
V_{CC} Standby Current CMOS	I_{SB1}	note 1		65	mA
V_{CC} Standby Current TTL	I_{SB2}	note 2		65	mA
Active Current (CMOS)	I_{CC1}	notes 1, 3			
—Commercial				130	mA
—Military				150	mA
Active Current (TTL)	I_{CC2}	notes 2, 3			
—Commercial				160	mA
—Military				180	mA
V_{PP} Supply Current	I_{PP}	$V_{PP}=V_{CC}$		100	µA
V_{PP} Read Voltage	V_{PP}	notes 1, 2	V_{CC}–0.4	V_{CC}	V
Input Load Current	I_{LI}	V_{IN}=5.5V or GND	–10	10	µA
Output Leakage Current	I_{LO}	V_{OUT}=5.5V or GND	–10	10	µA

Notes:
1. CMOS inputs: GND ± 0.3V or V_{CC} ± 0.3V.
2. TTL inputs: $V_{IL} \leq 0.8V$, $V_{IH} \geq 2.0V$.
3. Active current is an AC test and uses AC timing levels.

Table 15. AC Timing Levels

Inputs:	0 to 3V Reference 1.5V
Outputs:	0.4 to 2.4V

Table 16. AC Characteristics

Parameter	Symbol	12MHz[1] Min	12MHz[1] Max	16MHz[2] Min	16MHz[2] Max
CLOCK CYCLE					
Clock Time	t_{CK}	84		62.5	
Clock Pulse Width High	t_{CKH}	29		25	
Clock Pulse Width Low	t_{CKL}	29		25	
HOST READ CYCLE					
Read Cycle Time	t_{RC}	45		35	
Address to Data Valid	t_{ACC}		40		30
$\overline{CS}$ to Data Valid	t_{CS}		40		30
$\overline{CS}$ to Tristate	t_{CSZ}	0	45	0	35
HOST WRITE CYCLE					
Pulse width to $\overline{CS}$ and $\overline{WR}$ LOW	t_{PWL}	23		18	
Pulse width to $\overline{CS}$ and $\overline{WR}$ High	t_{PWH}	12		10	
Data setup to $\overline{WR}$	t_{SD}	5		5	
Data hold to $\overline{WR}$	t_{HD}	12		10	
RESET CYCLE					
$\overline{RESET}$ setup	t_{SR}	10		10	
$\overline{RESET}$ to tristate of ADD, HAD, HD, I/O	t_{RZ}	35		35	
$\overline{RESET}$ clocked to OUTCNTL low	t_{ROL}	35		35	
ADDRESS TIMING					
Address/Data setup	t_{SADD}	0		0	
Address/Data hold	t_{HADD}	8		8	
Clocked Counter to Address output	t_{CADD}		30		22
Clocked Address Register to Address	t_{RADD}		40		30
ADOE enable to data valid	t_{ADOE}		40		30
HADOE enable to data valid	t_{HADOE}		40		30
Address output disable	t_{CKZ}	0	45	0	35

4

Table 16. AC Characteristics (Con't)

Parameter	Symbol	12MHz[1] Min	12MHz[1] Max	16MHz[2] Min	16MHz[2] Max
DATA AND I/O TIMING					
Clock to I/O Output Valid	t_{CKIO}		30		25
Clock to HD Output	t_{CKHD}		35		30
I/O data setup	t_{SIO}	5		5	
I/O data hold	t_{HIO}	5		5	
HD data setup	t_{SHD}	5		5	
HD data hold	t_{HHD}	12		9	
HDOE enable to data valid	t_{HDOE}		40		30
Bus Output Disable	t_{CKZ}	0	45	0	35
TEST AND INTERRUPT TIMING					
Condition Code setup	t_{SCC}	60		50	
Condition code hold	t_{HCC}	0		0	
Clock to OUTCNTL Valid	t_{COV}	0	25	5	20
Minimum Interrupt pulse for acceptance	t_{IPWA}	15		10	
SPECIAL FUNCTION TIMING (I/O Bus)					
SQ15 Setup	t_{SSQ15}	15		12	
SQ15 hold	t_{HSQ15}	5		5	
SQ0 setup	t_{SSQ0}	15		12	
SQ0 hold	t_{HSQ0}	5		5	
Clock to Q0 output	t_{CKQ0}		35		30
Clock to Q15 output	t_{CKQ15}		35		30
Address counter enable setup	t_{SACEN}	15		10	
Address Counter enable hold	t_{HACEN}	0		0	
Block Counter enable setup	t_{SBCEN}	15		10	
Block Counter enable hold	t_{HBCEN}	5		5	
External output enable to data valid	t_{SFV}		30		25
External output enable to high impedance	t_{SFZ}		30		25

Notes:
1. Operating temperature range: Commercial, Industrial, Military
2. Operating temperature range: Commercial

Figure 21. Clock Cycle Timing

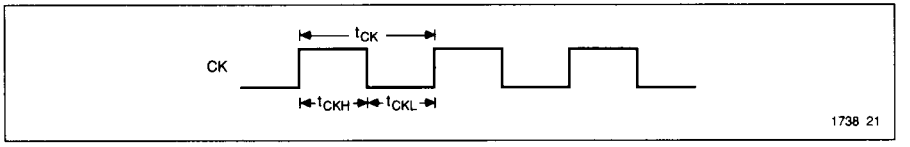


Figure 22. Host Read Cycle Timing

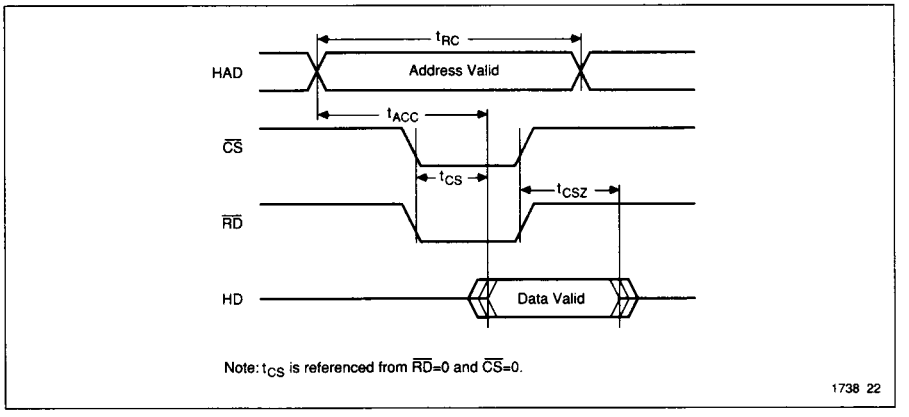


Figure 23. Host Write FIFO Cycle Timing

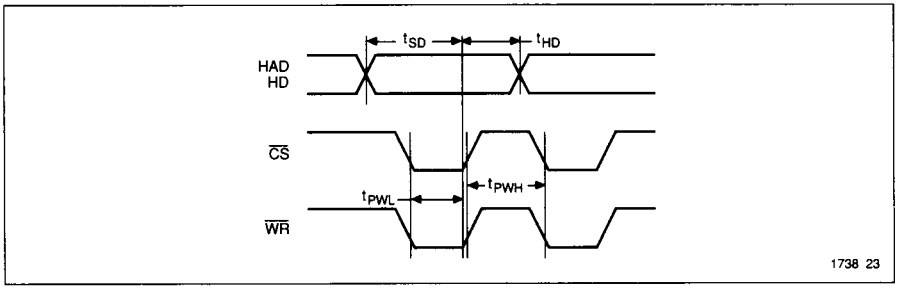
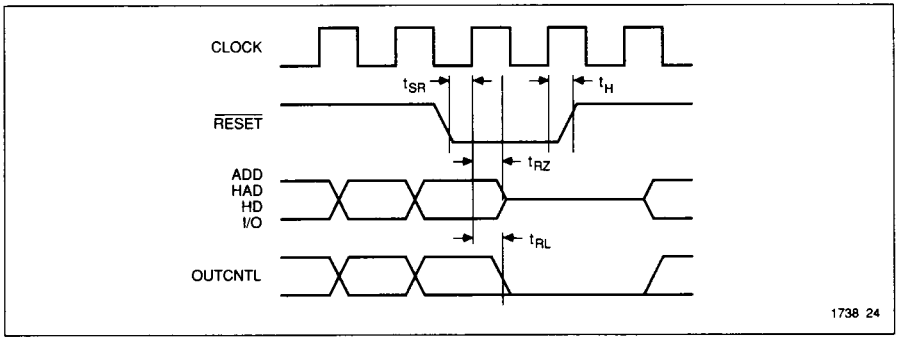
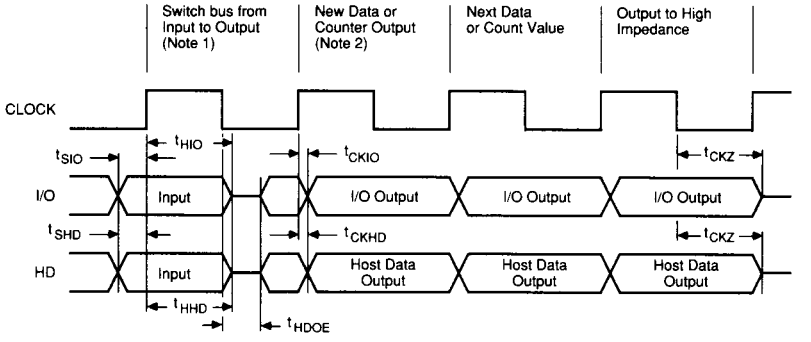


Figure 24. Reset Cycle Timing



4

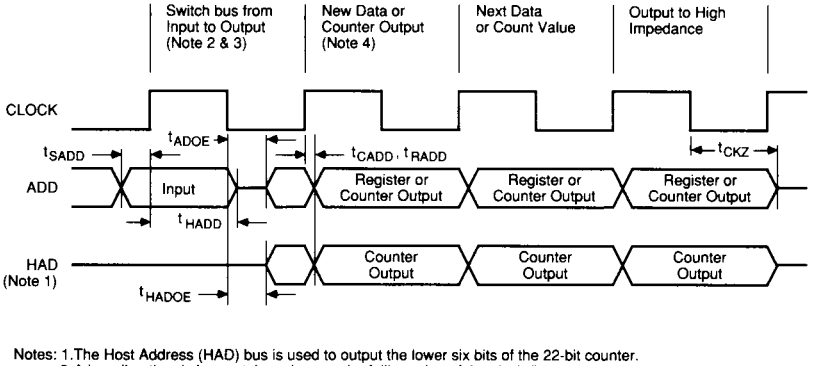
Figure 25. Data and I/O Timing



Notes: 1. A bus directional change (input-to-output or output-to-input) takes place on the falling edge of the clock.
2. New data or count value is latched on the rising edge of the clock.

1738 25

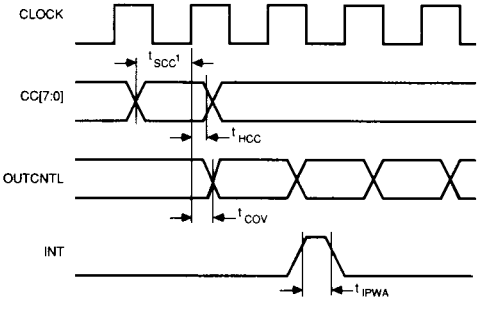
Figure 26. Address Timing



Notes: 1. The Host Address (HAD) bus is used to output the lower six bits of the 22-bit counter.
2. A bus directional change takes place on the falling edge of the clock (input-to-output or output-to-input).
3. Selection of the source to be output on a bus occurs on the falling edge of the clock (i.e., counter or address register).
4. New data or count value is latched on the rising edge of the clock.

1738 26

Figure 27. Test and Interrupt Timing



Note: 1. Since condition codes are not latched, they should be stable t_{SCC} prior to being tested.

1738 27

Figure 28. Special Function Timing

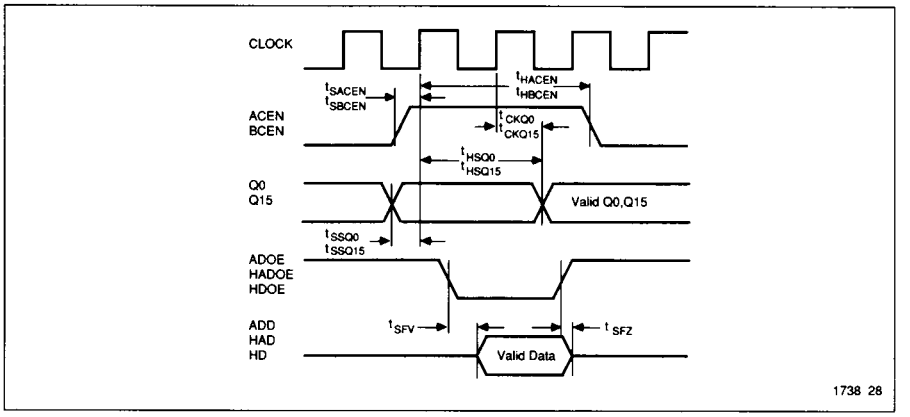
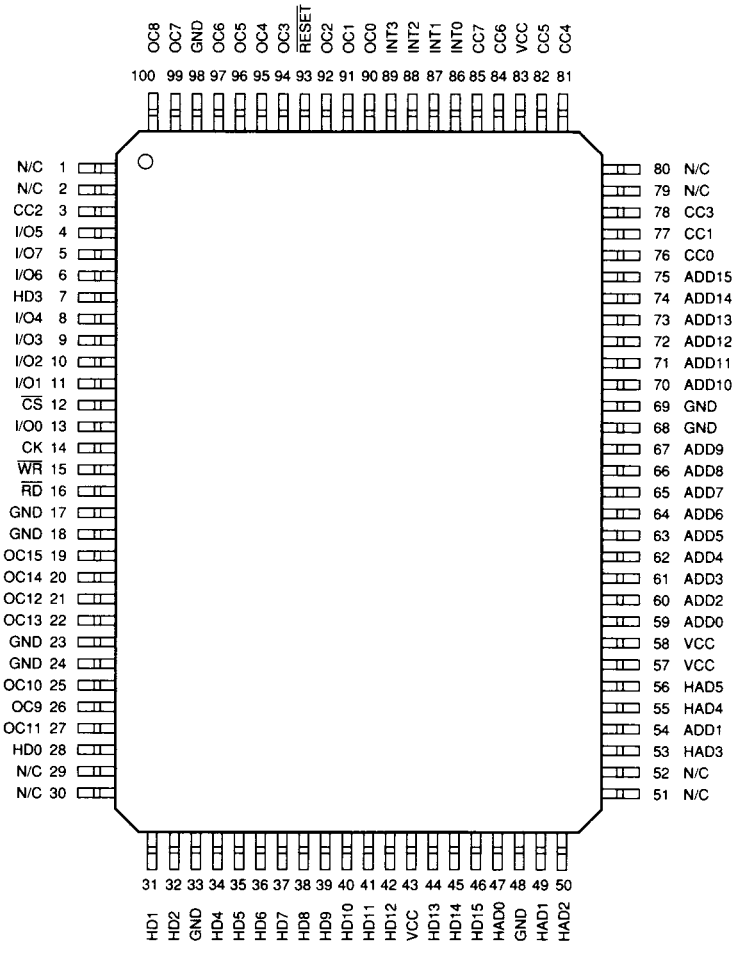
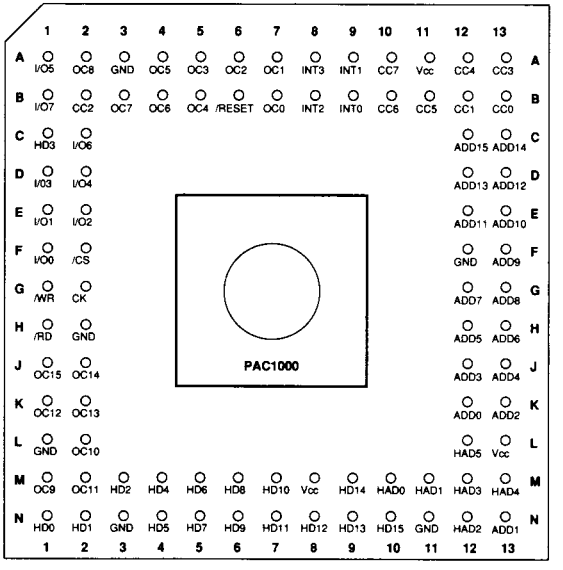


Figure 29. 100-Pin PQFP Pin Assignments

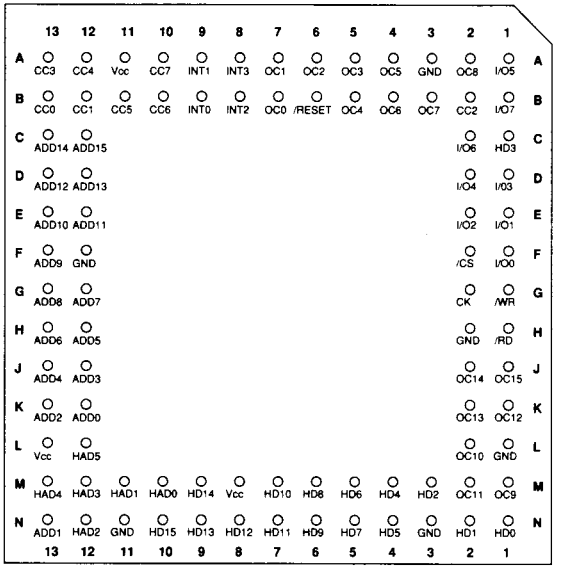


4

Figure 30. 88-Pin Ceramic PGA Pin Assignments



TOP (THROUGH PACKAGE) VIEW



BOTTOM VIEW

1738 29

Table 17. PGA Pin Assignments

Name	Pin	Name	Pin	Name	Pin
$\overline{CS}$	F2	GND	H2	I/O0	F1
$\overline{RD}$	H1	GND	L1	I/O1	E1
$\overline{RESET}$	B6	GND	A3	I/O2	E2
$\overline{WR}$	G1	GND	F12	I/O3	D1
ADD0	K12	GND	N3	I/O4	D2
ADD1	N13	GND	N11	I/O5	A1
ADD10	E13	HAD0	M10	I/O6	C2
ADD11	E12	HAD1	M11	I/O7	B1
ADD12	D13	HAD2	N12	INT0	B9
ADD13	D12	HAD3	M12	INT1	A9
ADD14	C13	HAD4	M13	INT2	B8
ADD15	C12	HAD5	L12	INT3	A8
ADD2	K13	HD0	N1	OC0	B7
ADD3	J12	HD1	N2	OC1	A7
ADD4	J13	HD10	M7	OC10	L2
ADD5	H12	HD11	N7	OC11	M2
ADD6	H13	HD12	N8	OC12	K1
ADD7	G12	HD13	N9	OC13	K2
ADD8	G13	HD14	M9	OC14	J2
ADD9	F13	HD15	N10	OC15	J1
CC0	B13	HD2	M3	OC2	A6
CC1	B12	HD3	C1	OC3	A5
CC2	B2	HD4	M4	OC4	B5
CC3	A13	HD5	N4	OC5	A4
CC4	A12	HD6	M5	OC6	B4
CC5	B11	HD7	N5	OC7	B3
CC6	B10	HD8	M6	OC8	A2
CC7	A10	HD9	N6	OC9	M1
CK	G2			VCC	A11
				VCC	L13
				VCC	M8

Instruction Set Overview

The PAC1000 architecture can perform three operations simultaneously in each instruction cycle. The operations are specified in the System Entry Language (PACSEL) using a single statement. PACSEL instructions can perform three operations:

- Program Control (PROGCNTL)
- CPU
- Output Control (OUTCNTL)

Each *instruction* is executed in a single cycle; the three *operations* are executed in parallel.

The syntax of a PACSEL statement has a label and three components:

```
[label:] PROGCNTL, CPU,
OUTCNTL;
```

The PROGCNTL component determines program flow and determines the next statement to be executed; the CPU component determines which operation is to be performed by the CPU; the OUTCNTL component determines the state of the control outputs.

A comma (,) is used to separate the instructions and a semi-colon marks the end of a statement. In general, each statement is executed in a single cycle.

In PACSEL statements, the PROGCNTL, CPU, OUTCNTL components can come in any order or any combination of Macro or Assembler operators. That is, you may mix Assembler operators among Macro operators. Tables at the end of this section summarize the Macro and Assembler operators.

In some cases, the same mnemonic is used to specify identical operations in both Macro and Assembler level.

You may:

- Specify all the components in the same statement in order to perform the operations in parallel:

  ```
  PROGCNTL, CPU, OUTCNTL;
  ```

- Specify components one at a time:

  ```
  CPU;
  PROGCNTL;
  OUTCNTL;
  ```

- Use components in any combination:

  ```
  PROGCNTL, CPU;
  PROGCNTL, OUTCNTL;
  CPU, OUTCNTL;
  ```

WSI recommends that, in general, you maintain a consistent ordering of these components and consistent groupings of Assembler-level and Macro operators, e.g. in separate files. This manual uses the PROGCNTL, CPU, OUTCNTL ordering.

When PROGCNTL is omitted, the implied instruction is CONTinue, that is, proceed to the next control instruction. When CPU is omitted, the implied instruction is NOP. When OUTCNTL is omitted, the implied instruction is MAINTain, that is, maintain the most recent OUTCTL, in Assembler order.

A summary of PACSEL Assembler and Macro statements follows.

Table 18. PACSEL Assembler Language Summary

Mnemonic	Arguments	Meaning
The PROGCNTL Operators		
ACSIZE	<16/22>	SET A COUNTER SIZE
CALL	<LABEL | LCPTR | FIFO>	UNCOND BRANCH SUBRTN
CALLC	<COND> <LABEL | FIFO>	COND BRANCH SUBRTN
CALLNC	<COND> <LABEL | FIFO>	INV COND BRANCH SUBRTN
CCASE	<CG> <VALUE>	BRANCH SUBRTN CASEBLK
CLI	<MASK>	CLEAR INTERRUPT
CONT(D)		CONTINUE
CPI	<VALUE>	PRIORITIZED SUB RTN
DI	<MASK>	DISABLE INTERRUPT
DSS		DISABLE SINGLE STEP MODE
EI	<MASK>	ENABLE INTERRUPT
ESS		ENABLE SINGLE STEP MODE
JCase	<CG> <VALUE>	UNCOND BRANCH CaseBLK
JMP	<LABEL | LCPTR | FIFO>	UNCONDITIONAL BRANCH
JMPC	<COND> <LABEL | FIFO>	CONDITIONAL BRANCH
JMPNC	<COND> <LABEL | FIFO>	INVERT COND BRANCH
JPI	<VALUE>	PRIORITIZED BRANCH
LDBP	<VALUE>	LOAD BP REG
LDBPD		LOAD BP COMP VALUE
LDLC	<VALUE>	LOAD COUNTER
LDLCD		LOAD CTR COMPUTED VAL
LOOPNZ	<LABEL>	REPEAT BRANCH,CNTRNZ
PLDLC	<VALUE>	PUSH VALUE & LDCTR
PLDLCD		PUSH VAL&LDCTR CM VL
POP		POP STACK
POPLC		POP STACK TO CNTR
PUSHLC		PUSH CNTR
RESTART		BRANCH TO 0
RET		RETURN
RC	<COND>	CONDITIONAL RETURN
RNC	<COND>	INV COND RETURN
RSTCON	<MASK>	RESET CONTROL REG
RSTIO	<MASK>	RESET I/O CONFIG REG
RSTMODE	<MASK>	RESET MODE REG
SETCON	<MASK>	SET CONTROL REG
SETIO	<MASK>	SET I/O CONFIG REG
SETMODE	<MASK>	SET MODE REG

Table 18. PACSEL Assembler Language Summary (Con't)

Mnemonic	Arguments	Meaning
The CPU Operators		
ADC	<ARG1> <ARG2> [<ARG3>]	ADD WITH CARRY
ADD	<ARG1> <ARG2> [<ARG3>]	ADD
AND	<ARG1> <ARG2> [<ARG3>]	BITWISE AND
CMP	<ARG1> <ARG2>	COMPARE
DEC	<ARG1> [<ARG2>]	DECREMENT
INC	<ARG1> [<ARG2>]	INCREMENT
INV	<ARG1> [<ARG2>]	INVERT
MOV	<DEST> <SRC>	MOVE SRC TODEST
NOP(D)		NO OPERATION
OR	<ARG1> <ARG2> [<ARG3>]	BITWISE OR
RDFIFO		READ FIFO DATA TO REG
SBC	<ARG1> <ARG2> [<ARG3>]	SUB WITH CARRY
SHLRQ	<REG> <RARG> <QARG>	SHIFT LEFT REG & Q
SHLR	<REG> <RARG>	SHIFT LEFT REG
SHRRQ	<REG> <RARG> <QARG>	SHIFT RIGHT REG & Q
SHRR	<REG> <RARG>	SHIFT RIGHT REG
SUB	<ARG1> <ARG2> [<ARG3>]	SUBTRACT
XOR	<ARG1> <ARG2> [<ARG3>]	EXCLUSIVE OR
XNOR	<ARG1> <ARG2> [<ARG3>]	EXCLUSIVE NOR
The MACRO Operators		
DIV	<ARG1> <ARG2> <ARG3>	DIVISION
MUL	<ARG1> <ARG2> <ARG3>	2'S COMP MULTIPLY
The OUTCNTL Operators		
MAINT(D)		MAINTAIN PREV VALUE
OUT	<VALUE>	OUTPUT

Table 19. PACSEL Macro Language Summary

The PROGCNTL Operators

```
ACSIZE <16/22>
CALL <label | LCPTR | FIFO> [ON] [NOT] [<cond>]
CASE n, PROGCNTL, CPU, OUTCNTL;
CLEAR <int level> [...<int level>]
CONFIGURE <pm1> [<pm2>...<pm10>]
CONT
DISABLE <int level> [<int level>...<int level>]
ELSE
ENABLE <int level> [<int level>...<int level>]
ENDFOR
ENDIF
ENDPSWITCH
ENDSWITCH
ENDWHILE
FOR <value>
GOTO <label | LCPTR | FIFO> [ON] [NOT] [<cond>]
IF [NOT] <cond>
INPUT <i/o pin> [<i/o pin>...<i/o pin>]
LOADBP <value>
OUTPUT <i/o pin> [<i/o pin>...<i/o pin>]
PRIORITY m, PROGCNTL, CPU, OUTCNTL;
PSWITCH
RESET <p1> [<p2>...<p10>]
RETURN [ON] [NOT] [<cond>]
SET <p1> [<p2>...<p10>]
SWITCH <case group>
WHILE [NOT] <cond>
```

Table 19. PACSEL Macro Language Summary (Con't)

The CPU-Operator Assignment

```
move
    <dest> := <src>
arithmetic expression
    <dest> := <arg1> <+/-> <arg2> <+/-> <arg3>
logical expression
    <dest> := <arg1> <logical operator> <arg3>
increment, decrement, invert, unary minus
    <dest> := <opr> <src>
macro expression
    <dest> := <arg1> [* | /] <arg2>
shift RAM
    <Rx> = Rx <shft opr> <shft arg>
shift RAM and q
    <QRx> = Q <shft opr> <shft arg> Rx <shft opr> <shft arg>
```

The OUTCNTL Operator

```
OUT <arg1> [<arg2>...<arg16>]
```

System Development Tools

PAC1000 System Development Tools are a complete set of PC-based development tools. They provide an integrated easy-to-use software and hardware environment to support PAC1000 development and programming.

The tools run on an IBM-XT, AT, PS2 or compatible computer running MS-DOS version 3.1 or later. The system must be equipped with 640 Kbytes of RAM and a hard disk.

Hardware

The PAC1000 System Programming Hardware consists of:

- WS6000 MagicPro Memory and PSD Programmer (XT, AT only)
- WS6010 Package Adaptor (88-Pin Ceramic Pin Grid Array) for the MagicPro Remote Socket Adaptor Unit
- WS6013 Package Adaptor (100-Pin PQFP) for the MagicPro Remote Socket Adaptor Unit

The MagicPro Programmer is the common hardware platform for programming all WSI programmable products. It consists of the IBM-PC plug-in Programmer Board and the Remote Socket Adaptor Unit.

Software

The PAC1000 System Development Software consists of the following:

- WISPER Software—PSD Software Interface
- IMPACT Software—Interface Manager for PAC1000
- PACSEL Software—System Entry Language
- PACSIM Software—Functional Simulator
- PACPRO Software—Device Programming Software

WISPER and IMPACT software provide a menu-driven user interface enabling other tools to be easily invoked by the user.

The system design is entered into PACSEL source program files using an editor chosen by the user. PACSEL supports assembly-level and high-level Macro programming.

The PACSEL Assembler produces object code format in single or multiple modules, which are then linked by the PACSEL Linker into a single load file with a format suitable for PACSIM and PACPRO.

The PACSIM functional simulator enables the user to test and debug programs by examining the state of PAC1000 internal registers before and during a complete functional simulation of the device.

PACPRO software programs PAC1000 devices by using the MagicPro hardware and the socket adapter.

The programmed PAC1000 is then ready to be used.

Support

WSI provides a complete set of quality support services to registered owners. These support services include the following:

- 12-month Software Updates.
- Hotline to WSI Application Experts—For direct design assistance.
- 24-Hour Electronic Bulletin Board—For design assistance via dial-up modem.

Training

WSI provides in-depth, hands-on workshops for the PAC1000 and the System Development Tools. Workshop participants will learn how to develop and program their own high-performance microcontrollers. Workshops are held at the WSI facility in Fremont, California.

Programming/ Erasing

Refer to the PAC1000 Users Manual found with the PAC1000-Gold and the PAC1000-Silver.

Ordering Information—PAC1000

Part Number	Speed (MHz)	Package Type	Package Drawing	Operating Temperature	Manufacturing Procedure
PAC1000-12Q	12	100-Pin Plastic Quad Flat Package	Q1	Commercial	Standard
PAC1000-12X	12	88-Pin Ceramic Pin-Grid Array	X1	Commercial	Standard
PAC1000-12XI	12	88-Pin Ceramic Pin-Grid Array	X1	Industrial	Standard
PAC1000-12XM	12	88-Pin Ceramic Pin-Grid Array	X1	Military	Standard
PAC1000-12XMB	12	88-Pin Ceramic Pin-Grid Array	X1	Military	MIL-STD-883C
PAC1000-16X	16	88-Pin Ceramic Pin-Grid Array	X1	Commercial	Standard

Ordering Information—System Development Tools

Part Number	Contents
PAC1000-GOLD	WISPER Software IMPACT Software PACSEL Software PACSIM Software PACPRO Software User's Manual WSI-Support WS6000 MagicPro Programmer One Socket Adaptor and Two PAC1000 Product Samples
PAC1000-SILVER	WISPER Software IMPACT Software PACSEL Software PACSIM Software PACPRO Software User's Manual WSI-Support
WS6000	MagicPro Programmer IBM PC plug-in Adaptor Card Remote Socket Adaptor
WS6010	88-Pin CPGA Adaptor Used with the WS6000 MagicPro Programmer
WSI-Support	Support Services, including: ❑ 12-month Software Update Service ❑ Hotline to WSI Application Experts ❑ 24-hour access to WSI Electronic Bulletin Board
WSI-Training	Workshops at WSI, Fremont, CA For details and scheduling, call PSD Marketing, (415) 656-5400

4